HIGH-RELIABILITY COMPUTER INTERFACE FOR WIRELESS INPUT DEVICES

Inventors: Martin George Morris, Vista, CA (US); Krishnasamy Anandakumar, San Diego, CA (US); Suresh Kumar Singamsetty, Carlsbad, CA (US); Dennis Ching Chung Kwan, San Diego, CA (US); Hock Thye Law, Carlsbad, CA (US)

Correspondence Address:
George O Saile and Assocs
28 Davis Ave
Poughkeepsie, NY 12603 (US)

Publication Classification
Int. Cl. G09G 5/08
U.S. Cl. 345/158

ABSTRACT

A computing system wherein communication between a host computer and peripheral units, e.g. computer mouse and keyboard, is performed using RF signals. The host computer and the peripheral units each contain a transceiver for managing and transmitting the RF communication messages. An acknowledgement is sent by the receiving transceiver to the sending transceiver to signify that the message was successfully received. If no acknowledgement is received by the sending transceiver, a subsequent RF communication message is sent until an acknowledgement is received. A sleep mode is invoked between messages to conserve battery power in the peripheral units, and the peripheral units send a report message to the host when waking periodically, or by a user demand, from the sleep mode signifying that the peripheral unit is active and ready to send or receive messages.
FIG. 2
Power Interface

CPU

ROM

RAM

MODEM

Baseband Hardware

Wake-Up Logic

USB

HOST

FIG. 3
Power Interface 430

CPU 402
ROM 404
RAM 406
MODEM 408
Baseband Hardware 409
Wake-Up Logic 410
Keyboard Scan 412

FIG. 4
FIG. 5
FIG. 6

Mouse Sleep Sequence

Device Side

Awaken due to Sleep Timer expired or Threshold reached

Message CRC OK—cancel ACK Timer; go to sleep

PC Side

Mouse Report

ACK Timer

ACK (w/o command)

Message CRC OK

SLEEP Timer

Mouse Report

ACK Timer

ACK (w/ command)

Message CRC OK

cancel ACK Timer

ACK Timer

ACK Timer

ACK Timer Expires; go to sleep

SLEEP Timer
Mouse Sleep Sequence
(with lost messages)

Device Side

PC Side

Awaken due to Sleep Timer expired or Threshold reached

Message CRC NC - discard

ACK Timer expires; back off

Backoff Timer Expires

Message CRC OK - cancel ACK Timer, go to sleep

Awaken due to Sleep Timer expired or Threshold reached

Message CRC OK - cancel ACK Timer, process command

Message CRC OK - cancel ACL Timer, duplicate XSN so just ACK

ACK Timer Expires; go to sleep

Mouse Report

ACK Timer

ACK (w/o command)

Backoff Timer

Mouse Report (retransmit)

ACK Timer

ACK (w/o command)

Message CRC OK

PC has no pending message for mouse, so just ACKs

Message CRC OK - duplicate XSN so ignore but ACK

Message CRC OK - just ACK

ACK Timer Expires; back off

ACK Timer expires; back off

ACK Timer

ACK (WITH command)

ACK Timer

ACK

ACK Timer

ACK

ACK Timer

Cancel ACK Timer

SLEEP Timer

FIG. 7
**FIG. 8**

- **MAC**
  - Write own device ID (for Rx)
  - RX_Enable = 1
- **RADIO**
  - Message received
  - processing latency
  - Message processed, ACK prepared
  - Set up registers
  - Write long sequence (for Tx)
  - Tx_Enable = 1
  - Send
  - Transmission time
  - Send Complete
  - Tx_Enable = 0

**FIG. 9**

- Rest of World
- Europe only
- HIGH RATE
  - RSSI>H, reset encryption
  - ACK Timeout & RSSI=L
- MEDIUM RATE
  - RSSI=L, reset encryption
- LOW RATE
  - Tx Successful: select next channel, reset encryption
  - ACK timeout: select next channel, reset encryption
  - ACK timeout: select next channel, reset encryption
  - RSSI=H, reset encryption
  - ACK Timeout & tried all 5 channels, reset encryption
1200 Pairing Sequence

Device Side

- User pushes pairing button on device side: listen for IDpair.
- CRC OK, extract PC side ID, respond to that ID with device ID.
- CRC bad, ignore.
- ACK timeout, back off.
- CRC OK, cancel ACK Timer, switch to listening for own ID.
- CRC OK, cancel ACK Timer, device now ready & idle.

PC Side

- User pushes pairing button on PC side: send request using IDpair with PC side device ID.
- ACK Timer.
- Pairing Request.
- ACK with Pairing Accept.
- ACK Timer.
- ACK with Pairing Accept (retransmit).
- ACK Timer.
- ACK (retransmit).
- ACK Timer.
- ACK Timer.
- ACK Timer.
- ACK Timer.
- Reset.
- ACK Timer.
- ACK.

- Repeat until response received or have tried N=1000 times.
- CRC OK, cancel ACK Timer, send ACK using IDpair.
- CRC OK, duplicate XSN; ignore but resend ACK using IDpair.
- CRC OK, cancel ACK Timer & send ACK using new device ID.

FIG. 12
FIG. 13
There is a message need to be transmitted

Enable RF receiver

Wait for 150us for PLL Locking

channel = busy

Check CCA report from RF

Channel = clear

Wait for a window T, then check CCA report from RF

Channel = clear

Switch to transmit mode to transmit

After finishing transmission, switch to RX mode and wait for the ACK

ACK timeout

Yes

Exceed maximum number of retransmission?

No

Wait for new message

Done

Received ACK

Random Backoff

FIG. 14
Set CAA threshold level (TH) to default value of M dBm

Measure PER and back-off rate (for N packets)

\[
\begin{align*}
\text{Back-off rate} & > z\% \land
\text{PER} < a\% \\
\text{Increase CAA threshold level} \\
TH & = TH + 6 \text{ dB}
\end{align*}
\]

\[
\begin{align*}
\text{Back-off rate} & < y\% \land
\text{PER} > b\% \\
\text{Decrease CAA threshold level} \\
TH & = TH - 6 \text{ dB}
\end{align*}
\]

**FIG. 15**

**FIG. 16**
Last Channel

RSSI = Low
High Data Rate

RSSI = Low & no message received (reset encryption)

RSSI = High & no messages for K seconds (reset encryption)

Medium Data Rate

No Message received
Listen for N seconds

Received message

Go to next channel and reset encryption

Listen for M seconds

Stay in that channel

If 5 channels are already cycled through and no message has been received. The transceiver goes back to HDR (reset encryption) and listens for K seconds. If no message is received within that K seconds and RSSI is still high, transceiver goes to MDR again.

FIG. 17
If RSSI is low, go to medium data rate and reset encryption.

If RSSI is high, go back to HDR (reset encryption) and try to make the link for T seconds. If no ACK is received and RSSI is still high, go to MDR again.
FIG. 21
FIG. 22

Start

Initialize

Patch EEPROM?

Yes

Copy EEPROM to RAM

No

Resume ROM Application

End

2300

2400

Start

MyFnPtr = null?

Yes

Call Substitute Function

No

Run Original Function

Return

FIG. 23

FIG. 24
HIGH-RELIABILITY COMPUTER INTERFACE 
FOR WIRELESS INPUT DEVICES

RELATED PATENT APPLICATION


[0004] This application is related to U.S. patent application docket number JA05-002, Ser. No. ______, filed on ______, assigned to a common assignee.

FIELD OF THE INVENTION

[0005] The present invention relates generally to computer systems and, more particularly, to an interface between a computer and input devices in communication with the computer over wireless links.

BACKGROUND OF THE INVENTION

[0006] Various computers and microprocessor-based devices and systems provide one or more user input devices to allow a user to control certain operations. Such an input device may be separated from the host computer or device and thus a communication link and an interface may be implemented to support proper communications between the input device and the host computer or device. Generally, each of the input device and the host computer/device includes appropriate software and hardware, for the communication link and interface.

[0007] For example, a typical desktop or laptop computer may have a keyboard and a pointing device for a user to input data or commands for controlling or operating the computer. Examples of the pointing device for computers include a mouse, a touch pad, a trackball, and a pointing stick (IHM laptops). In addition to keyboards and pointing devices, examples of some other user input devices include joysticks and game pads for computers and microprocessor-based game machines, control units for other microprocessor-based devices. In general, a user uses an input button, a control stick, one key or a key combination, or a combination thereof to input data or a command. Circuitry in the input device converts the input data or command into a proper form for transmitting to the computer or device.

[0008] Such an input device generally uses a particular communication link to transmit the input data or command to the computer or device. An input device may be a wireless input device using a wireless communication link or a wired link using an electrical cable. Input devices with wired links may be implemented based on the PS/2 keyboard interface, the USB 1.0 and USB 2.0 interfaces, and other interfaces. The wireless communication link may be implemented by a radiation transmitter to send the input to a corresponding radiation receiver at the computer or device. Many wireless input devices use RF radiation links based on different radio interfaces such as IEEE 802.11g for low speed links and wireless USB 2.0 and IEEE 1394 for relatively high speed links. Some of these wired or wireless input devices may use the Human Interface Device (HID) protocol over wired or wireless USB links or other non-USB communication links.

[0009] Wireless input devices beneficially increase the flexibility of the interaction between a user and a host computer in that no wired connection is required with the host computer. However, given that a wired connection generally provides a source of power for an input device, wireless input devices are required to be self-powered (e.g., battery-powered). Unfortunately, batteries used to power existing wireless input devices typically last for a period of time significantly less than the useful life of such devices. As a consequence, the convenience and value of such devices are diminished as a consequence of the need for regular battery replacement.

[0010] Existing wireless input devices are also frequently of limited range and the wireless link established for communication with the host computer is often rather unreliable and/or exhibits a high latency. In addition, such wireless links are often relatively insecure and thus susceptible to eavesdropping or unauthorized monitoring.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] This invention will be described with reference to the accompanying drawings, wherein:

[0012] FIG. 1 is a diagram of a computer system of the present invention incorporating a wireless input interface;

[0013] FIG. 2 is a diagram of the layered architecture of the present invention for the wireless unit contained in the host computer;

[0014] FIG. 3 is a block diagram of the wireless unit of the present invention;

[0015] FIG. 4 is a block diagram of the present invention depicting an exemplary embodiment of the wireless unit;

[0016] FIG. 5 is a block diagram of the present invention depicting an exemplary embodiment of the wireless unit;

[0017] FIG. 6 is an event trace diagram of the present invention for mouse sleep sequence;

[0018] FIG. 7 is an event trace diagram of the present invention for mouse sleep sequence with lost messages;

[0019] FIG. 8 is an event trace diagram of the present invention for enabling of the transmitter/receiver within the RF units;

[0020] FIG. 9 is a state transition diagram of the present invention demonstrating the manner in which the data rate and channel of the wireless unit are changed;

[0021] FIG. 10 is an event trace diagram of the present invention for normal keyboard sequence;

[0022] FIG. 11 is shown a computer system of the present invention;
FIG. 12 is an event trace diagram of the present invention demonstrating the messages exchanged during a pairing operation;

FIG. 13 is a state transition diagram of the present invention;

FIG. 14 is a flowchart of the present invention of a carrier sense/clear channel assessment procedure;

FIG. 15 is a flowchart of the procedure for dynamically adjusting the CCA threshold level used by the RF unit;

FIG. 16 is a high-level flow diagram representative of a general approach to interference handling consistent with the present invention;

FIG. 17 is a flow diagram of the present invention for the method in which a host transceiver converts between HDR and MDR modes;

FIG. 18 is a flow diagram of the present invention for the method in which a device transceiver converts between operations within the HDR and MDR modes;

FIG. 19 is a timing diagram of the present invention for the transition of the host transceiver into and out of “awake” and “sleep state” operation;

FIG. 20 is a state transition diagram of the present invention for operation of the host transceiver when in Suspend mode;

FIG. 21 is a state transition diagram of the present invention depicting the modifications to the messaging protocol of the mouse transceiver upon entry of the host transceiver into Suspend mode;

FIG. 22 is a high-level diagram of the present invention showing the incorporation of EEPROM in the wireless transceivers;

FIG. 23 is a flow diagram of the present invention for the inclusion of a patch EEPROM in the wireless transceivers; and

FIG. 24 is a flow diagram of the present invention for substituting a new function for a default function.

Detailed Description of the Invention

System Overview

Turning now to FIG. 1, a computer system 100 incorporating a wireless input interface in accordance with the invention includes a host computer 102, a wireless keyboard 104 and a wireless pointing device or “mouse” 106. As illustrated in FIG. 1, the wireless keyboard 104 and the wireless mouse 106 are in communication with the host computer 102 over wireless communication links 108 and 110, respectively. As shown, the host computer 102 includes or is attached to a wireless unit 112 through which the communication links 108 and 110 are respectively established with a wireless unit 114 within the wireless keyboard 114 and a wireless unit 116 of the wireless mouse 106. The wireless unit 112 may be built into the chassis of the host computer 102 or added as an external peripheral device in the form of, for example, a “dongle”. As an external peripheral device, the wireless unit 112 may either be interfaced to the computer 102 through a wired USB connection or by other available means such as the PS/2 keyboard connector.

During operation of the system 100, the wireless keyboard 104 and the wireless mouse 106 interact with the host computer 102 via wireless communication links 108 and 110. In particular, the wireless unit 112 receives keystroke and other data originating from the wireless keyboard 104 and the wireless mouse 106 over wireless communication links 108 and 110 and passes it to the host computer 102 in such a way that the host computer 102 is unaware of the existence of the wireless links 108 and 110.

As is described hereinafter, the wireless units 112, 114 and 116 of the present invention are configured to enable the communication links 108 and 110 to exhibit low latency and high reliability relative to conventional approaches employed using wireless peripheral devices. In addition, given that the wireless keyboard 104 and wireless mouse 106 will generally be battery powered devices, minimization of power consumption will typically be a primary design consideration. As a consequence, the wireless units 114 and 116 respectively incorporated within the wireless keyboard 104 and the wireless mouse 106 are disposed to cycle among various power-saving modes so as to conserve battery power and thereby substantially reduce the frequency of required battery replacement operations.

High-Level Architecture of Wireless Units

FIG. 2 illustrates a view of the layered architecture 200 of the wireless unit 112, it being understood that in the exemplary embodiment the wireless units 114, 116 are of essentially the same general architecture. As will be apparent to those skilled in the art, the layers depicted in FIG. 2 may be realized in hardware, firmware, or as software instructions stored on a computer-readable medium. Referring to FIG. 2, the unit 112 is seen to include a physical layer 204, a media access control (MAC) layer 208, and a device interface layer 212. As shown, the physical layer 204 interfaces with an antenna element 220.

The device interface layer 212 may be implemented as any known interface permitting the wireless unit 112 to interface with the host computer 102. Such a known interface may be designed to support communications between the host computer 102 and the wireless unit 112 in accordance with a standard communications protocol. For example, the device interface 212 may be designed to serve as a USB, PS2 or GPIO interface.

As is described below, the MAC layer 208 serves to control access to the wireless communication links 108, 110. That is, MAC layer 208 is responsible for enabling data to be transferred between the device interface 212 and the physical layer 204, and vice-versa. As shown, a portion of the functions associated with the MAC layer 208 in the exemplary embodiment are carried out by baseband hardware 260, but this is certainly not required. The physical layer 204 may comprise any structure or collection of elements functioning to transmit and receive bits of data over the wireless communication links 108, 110. As shown, the physical layer 204 includes a radio interface portion 250, which represents the registers and signals that are used to transfer messages between the physical layer 204 and the MAC layer 208. One potential implementation of the phys-
cal layer 204 is described in, for example, the above-referenced provisional application filed on even date here-

[0042] FIG. 3 illustratively represents a block diagram of the wireless unit 112 of FIG. 1 according to an exemplary embodiment of the present invention. As shown, the wireless unit 112 (also referred to herein as the host transceiver 112) includes a CPU 302 that is coupled via a CPU bus (not shown) to ROM 304, RAM 306, a modem 308, baseband hardware 309, wake-up logic 310, USB 312, and an input/output (I/O) module 316. The wake-up logic in the host device, although it is not generally required in this application, is an integral part of the logic implementation, which is the same in both the computer and peripheral input device. Also shown coupled to the modem 308 is an RF unit 314, which is coupled to an antenna 220. A power interface portion 330 provides power regulation to both analog and digital components of the host transceiver 112. In one embodiment, the host transceiver 112 is realized as a single system-on-chip IC with protocol stack and application software being integrated in built-in mask ROM, but this is certainly not required.

[0043] In the exemplary embodiment, when the host transceiver 112 is transmitting information to one of the wireless devices 114, 116, the information is first encrypted, formatted and protected with a cyclical redundancy check (CRC) by the baseband hardware 309. The modem 308 then receives and encodes (e.g., with differential binary phase shift key (DBPSK) encoding) the formatted, encrypted and CRC protected information before it is up-converted for transmission by the RF portion 314.

[0044] When receiving a signal from one of the devices 114, 116, the RF unit 314 down converts the received signal to an intermediate frequency (IF), and converts the IF frequency signal to a digital IF signal. The modem 308 then decodes the digital IF signal and checks the CRC to regenerate the original encrypted information, which is then decrypted by the baseband hardware 309.

[0045] With respect to transmitting and receiving data, the modem 308 in the exemplary embodiment has four different modes: a high data rate (HDR) mode; a medium data rate (MDR) mode; a low data rate (LDR) mode and a spread mode. The HDR mode is the default mode, which can provide, for example, 150 kbps data transmission. The data rates for the MDR, LDR and spread mode are 30 kbps, 10 kbps and 13.64 kbps respectively. As described further herein, spread mode is used when there is interference from similar wireless device(s) (e.g., other host and device transceivers), and MDR is used when there is strong interference (e.g., narrowband interference) such as from citizen band (CB) radio.

[0046] In the exemplary embodiment, the host transceiver 112 is able to detect interference and switch to appropriate modes automatically. As a consequence, the host transceiver 112 provides highly reliable wireless data transfers even in an environment with multiple other wireless users. In the exemplary embodiment, the LDR mode is for European compliance purposes and may be omitted in transceivers intended for non-European markets. The data transmission rates of the exemplary embodiment (i.e., 150 kbps, 30 kbps, 10 kbps and 13.64), are more than sufficient for typical manual input devices (e.g., the keyboard 104 and mouse 106), with very little or no perceptible latency.

[0047] The RF portion 314 in the exemplary embodiment operates to transmit and receive signals in accordance with the operating mode (i.e., the HDR, MDR, LDR and spread mode) of the modem 308. In MDR mode for example, the RF portion 314 supports multiple selectable transmit frequencies so data may be selectively transmitted over a frequency channel that is substantially free from a strong narrowband interferer such as a citizens band (CB) radio.

[0048] In the exemplary embodiment, the I/O unit 316 of the host transceiver 112 is programmable to allow general-purpose I/O pins (not shown) of the host transceiver 112 to be selectively dedicated to a variety of interface communication protocols for communication with the host computer 102. As shown in FIG. 3 for example, the I/O unit 312 is programmable so as to allow the following five I/O communication protocols to be selectively used with the general-purpose I/O pins: a general purpose input/output (GPIO) 318, an intelligent interface controller (IIC) path 320, a universal asynchronous receiver/transmitter interface (UART) 322, a USB interface 324 and a bidirectional synchronous serial interface (PS/2) 326.

[0049] The IIC interface 320 is a two-wire, bi-directional serial bus which provides a simple method of data exchange between devices. In the exemplary embodiment, the IIC interface is used for downloading executable programs from external EEPROM to RAM 306 (e.g., to change functionality of certain aspects of the host transceiver), and/or reading configuration parameters that are stored in external EEPROM. In one embodiment, e.g., when the CPU clock is 12 MHz the clock speed for the IIC interface is software programmable from 200 Hz to 400 KHz. When the CPU clock is 12 MHz the host transceiver 112 may either be selected (e.g., via software) to be a master or a slave device.

[0050] The UART interconnect 322 provides serial communications between the host transceiver and terminal equipment (e.g., the host computer 102). In one embodiment, the baud rate is software programmable from 250 bps to 330 Kbps. The Universal Serial Bus (USB) interface 324 is a personal computer (PC) interconnect that can support simultaneous attachment of multiple devices. The USB module 312 in the present embodiment is realized by dedicated hardware and includes a USB function controller (not shown) and a full speed (12 Mbits/s) USB transceiver (not shown). The PS/2 interface 326 is a two-wire (DATA, CLOCK), bi-directional synchronous serial interface. The PS/2 interface 326 in one embodiment includes two PS/2 interfaces: one for communications with the keyboard 104 and the other for communications with the mouse 106.

[0051] In one embodiment, dedicated hardware in the host transceiver 112 is associated with one or more of the above described communication protocols. Although it is not necessary to dedicate hardware for I/O communications, latency may be substantially reduced over alternative CPU-driven software implementations.

[0052] Referring next to FIG. 4, shown is a block diagram depicting an exemplary embodiment of the wireless unit 114 of FIG. 1. As shown, the wireless unit 114 (also referred to herein as the keyboard transceiver 114) includes a CPU 402 that is coupled to ROM 404, RAM 406, a modem 408, baseband hardware 409, RF unit 414 and an I/O module 416, which in the exemplary embodiment, are substantially the
same as the corresponding functional components within the host transceiver 112. Also shown are wake-up logic portion 410 and a keyboard scan module 412.

[0053] As shown, four exemplary communication protocols are available for the keyboard transceiver 114 to communicate with other devices: a general-purpose input/output (GPIO) 418, an intelligent interface controller (F/C) path 420, a universal asynchronous receiver/transmitter interface (UART) 422, and a keyboard interface 424. These protocols and interfaces are made available to support various design options for the keyboard.

[0054] The keyboard interface 424 in one embodiment is realized with 20 GPIO ports that are dedicated to 20 corresponding columns of a keyboard’s bare key switch contacts and 8 GPIO ports that are dedicated to 8 corresponding rows of the keyboard’s bare key switch contacts. In addition, three optional high-drive open-drain outputs are made available to support up to three LED devices on the keyboard. In this embodiment, the I/O module 416 is programmed to switch the 28 GPIO ports dedicated to the keyboard to the keyboard scan module 412.

[0055] The keyboard scan module 412 detects key presses and releases by receiving inputs from the keyboard interface 424 and performing debouncing and rollover handling. Debouncing is performed by keeping an image of the keyboard state in memory for the last N1 (e.g., three) scan cycles. In the exemplary embodiment, the keyboard scan module 412 does not report a state change until it persists for N1 scan cycles (the scan rate is approximately N2 (e.g., four) milliseconds per scan). As a consequence, the debounce time is approximately N1*N2 milliseconds. In one embodiment, the values of N1 and N2 may be changed via the host transceiver 112 by updating EEPROM of the host transceiver 112 with new values. The host transceiver 112 then sends the updated information to the keyboard transceiver 114 in a configuration message when communication is established with the host transceiver 112.

[0056] In operation, each time a key press or release is detected, the keyboard scan module 412 provides key code (i.e., column and row) information to the CPU 402, and the CPU 402 generates a message indicating the row and column. The message with row and column information is transferred from the keyboard transceiver 114 to the host transceiver 112. The host transceiver 112 receives the message and then maps the row and column data into key codes, macros, or special functions.

[0057] After a period of inactivity, the CPU 402 instructs the wake-up logic portion 410, as described further herein, to place the keyboard transceiver 114 in a sleep mode. The wake-up logic portion 410, in combination with the power interface 430, then effectively shuts down the CPU 402 by depriving it of a clock signal. In addition, a scan oscillator (not shown) in the keyboard scan module 412 is also deactivated so that the keyboard scan module 412 no longer carries out the keyboard scanning described above. Instead, the row inputs to the keyboard scan module 412 are logically ORed together so that any key-press will trigger the keyboard scan module 412 to restart the scanning process.

[0058] When the keyboard scan module 412 (operating in sleep mode) detects a key press, it sends a key press notification signal to the wake-up logic portion 410, which, in combination with the power interface 430, brings the keyboard transceiver 114 out of sleep mode by reactivating the clock signal to the CPU 402. Additional details of communications between the keyboard transceiver 114 and the host transceiver 112 when the keyboard transceiver 114 enters and exits sleep mode are described in the above-referenced provisional application filed on even date hereafter.

[0059] Referring next to FIG. 5, shown is a block diagram depicting an exemplary embodiment of the wireless unit 116 of FIG. 1. As shown, the wireless unit 116 (also referred to herein as the mouse transceiver 116) includes a CPU 502 that is coupled to ROM 504, RAM 506, a modem 508, baseband hardware 509, RF unit 514, and the I/O module 516, which in the exemplary embodiment, are substantially the same as the corresponding functional components within the keyboard transceiver 114. Also coupled to the CPU 502 is a wake-up logic portion 510, which is in communication with a mouse scan module 512.

[0060] As shown, four exemplary communication protocols are available for the mouse transceiver 114 to communicate with other devices: a general-purpose input/output (GPIO) 518, an intelligent interface controller (F/C) path 520, a universal asynchronous receiver/transmitter interface (UART) 522, and a mouse interface 524. These protocols and interfaces are made available to support various design options for the mouse.

[0061] In the exemplary embodiment, the mouse transceiver 116 receives, via a mouse interface 524, motion signals from mouse motion transducer 510, which is configured and positioned within the wireless mouse 106 to convert motion of the wireless mouse 106 into the motion signals.

[0062] Advantageously, the configuration of the mouse interface 524 in this embodiment is selectable to conform to the communication protocol of the mouse motion transducer 524, which may vary depending upon the manufacturer and the type of technology (e.g., mechanical or optical position tracking) utilized by the wireless mouse 106. Specifically, the I/O module 516 is programmable so that GPIO pins of the mouse transceiver 116 (not shown) are dedicated for communications in accordance with the protocols utilized by the wireless mouse 106.

[0063] In one embodiment for example, the mouse interface 524 is configured to communicate as an optical mouse interface according to a secure digital I/O communication protocol (SDIO), which uses an F/C-like read/write sequencing scheme in which the mouse transceiver 116 operates as the master and the wireless mouse 106 as the slave. This configuration may be used, for example, to communicate with Agilent™ wireless mouse devices with SDIO interface capability including Agilent™ device No. ADNS-2030.

[0064] In another embodiment, the mouse interface 524 is configured to communicate as an optical mouse interface according to serial peripheral interface (SPI) protocols. In this embodiment, the mouse interface 524 includes four signals: a clock (CLK), a slave output (SO), a slave input (SI) and a slave select (CS), and the mouse transceiver 116 operates as the master while an optical sensor in the mouse 106 operates as the slave.
When the wireless mouse 106 utilizes mechanical position tracking technology, the mouse interface 524 includes one pair of quadrature signals for each of the X, Y, and Z axes, wherein the X and Y axes are associated with the translational movement of the wireless mouse 106 and the Z-axis is associated with movement of a roller ball of the wireless mouse 106.

In the exemplary embodiment, the mouse scan module 512 operates in either a mechanical mode or an optical mode depending upon whether the wireless mouse 106 utilizes mechanical or optical position tracking. When operative in the mechanical mode, a single-axis state machine of a type described in the above-referenced pending patent application is executed by the mouse scan module 512 with respect to each of three perpendicular directional axes (i.e., the X, Y and Z axes). Operation in the mechanical mode also relies upon a button press detector (not shown), which preferably implements “de-bouncing” in the same manner as was described above with reference to the keyboard scan module 512. Although the button press detector operates substantially identically in the mechanical and optical modes, in the exemplary embodiment the mouse scan module 512 does not execute state machines during operation in the optical mode. Instead, the mouse scan module 512 sends serial messages to the wireless mouse 106 and receives back position difference information or “deltas”. These position deltas replace the counter values utilized during mechanical mode operation within the mouse position reports generated by the mouse scan module 512.

Operation of Mouse and Keyboard Wireless Units

Transition Among Operative States

As mentioned above with reference to FIG. 2, the MAC layer 208 is responsible for formatting messages conveyed by the MAC layer 208 to and from the device interface 212 and the physical layer 204. This aspect of the MAC layer 208 may be appreciated with reference to FIGS. 6, 7 and 10, which are event trace diagrams representative of various messages exchanged between the host wireless unit 112 and the keyboard/mouse wireless units 114, 116 under the control of their respective MAC layers 208.

Referring initially to FIG. 10, an event trace diagram 1000 is provided which depicts the message sequences associated with normal operation of the keyboard/mouse wireless units 114, 116. In addition, FIG. 10 also illustrates a message sequence between the mouse wireless unit 116 and the host wireless unit 112 in the case in which a message sent by the mouse wireless unit 116 is lost or corrupted and re-transmitted.

Turning now to FIG. 6, there is shown an event trace diagram 600 representative of an event sequence pursuant to which the wireless unit 116 for the wireless mouse 106 transitions into and out of sleep state under the control of its MAC layer 208. In the case in which the wireless unit 116 is awakened due to expiration of this time-out interval, the counters associated with each of the three directional axes are compared against a predefined value (e.g., 2). If none of the counters exceed this threshold, the time-out is restarted and sleep resumed; otherwise, a mouse report message is sent to the host transceiver 112 and these counters are reset to zero. As is shown in FIG. 6, once receipt of this mouse report message is acknowledged by the host transceiver 112, the wireless unit 116 returns to sleep state. If an acknowledgment is not received, the mouse report message is discarded and the wireless unit 116 returns to sleep state.

FIG. 7 illustrates, in a manner similar to FIG. 6, an event trace diagram 700 which represents an event sequence pursuant to which the wireless unit 116 for the wireless mouse 106 transitions into and out of sleep state under the control of its MAC layer 208 in the case in which messages are lost between the host wireless unit 112 and the wireless unit 116.

Consistent with the invention, the MAC layer 208 is also responsible for the overall transfer of data between the device interface 212 and the physical layer 204 and controls the transition of the applicable wireless unit 112, 114, 116 into and out of various operative states. Specifically, the MAC layer 208 of a given wireless unit is disposed to govern its transition among sleep, listen, backoff and pending states in the manner described hereinafter.

Turning now to FIG. 13, there is shown a state transition diagram 1300 representative of the manner in which transitions are made between the various states of a state machine executed by the MAC layer *208. In general, whenever one of the wireless units 114, 116 receives a message containing a long sequence (described below) identifying such unit 114, 116, its MAC layer 208 checks the CRC to decide if the message is valid. A message received from the host wireless unit 112 (other than an ACK only) is acknowledged immediately. This acknowledgement may, but need not be, accompanied by data intended for the host wireless unit 112. If a wireless unit 114, 116 sends data with or without a “piggyback” acknowledgement, the host wireless unit 112 is disposed to acknowledge the new data.

As is indicated by FIG. 13, if data is sent by a wireless unit 114, 116 and an acknowledgement is not immediately received following a short “ACK time-out” period, it is retransmitted by the wireless unit 114, 116 after waiting a “BACKOFF time-out” intended to break deadlocks caused by simultaneous transmissions. The “ACK time-out” period is selected to be roughly equal to the sum of (i) the round trip message transfer time between the applicable device (i.e., the wireless keyboard 104 or wireless mouse 106) and the host computer 102, and (ii) an allowance for processing time. In this regard an ACK time-out period of approximately 1.2 ms has been found to be appropriate for a number of anticipated implementations. It is observed that the time-out values described herein as being associated with the MAC layer 208, including the above-described ACK time-out period, are dependent upon the data rate used for the relevant transmission. In particular, the values provided herein assume operation in a the HDR mode (i.e., 150 kb/s) discussed above. For transmissions occurring during operation in the MDR mode (i.e., 30 kb/s), the provided
timeout values should be quintupled. Finally, the provided timeout values should be multiplied by a factor of 15 in connection with operation in the LDR mode (i.e., 10 kbps).

[0074] In the exemplary embodiment the BACKOFF timeout is randomized between wireless units. Such randomization may be affected by, for example, utilizing the device ID of the applicable wireless unit in generating the period of the BACKOFF timeout. If a wireless unit is required to delay for the period of an additional BACKOFF timeout in connection with a given transmission, the duration of the BACKOFF timeout may be increased linearly in accordance with the device ID. Specifically, in an embodiment where N corresponds to the number of consecutive ACK timeouts for a given data packet, and IDx is the device ID of the transmitting wireless unit, the duration of the BACKOFF timeout is given as BACKOFF\text{min}+(N\cdot f(IDx)). In this embodiment BACKOFF\text{min} is approximately 0.5 ms, and f(IDx) is chosen such that the average extension per consecutive BACKOFF timeout increases by approximately 1 ms.

[0075] In order to avoid or minimize any confusion resulting from retransmission of the same data, each message preferably contains a one-bit "transmit sequence number" which is incremented (toggled) for each new transmission and maintained the same for a retransmission. To avoid similar difficulties from arising in connection with lost acknowledgements, a one-bit "receive sequence number" is incremented whenever a new packet is acknowledged by a wireless unit. This bit indicates the sequence number expected in the next packet from the transmitting wireless unit, and thus effectively serves as an acknowledgment of receipt of the prior packet. In other words, if a wireless unit successfully receives a packet numbered "zero", the receive sequence number is set to "1" in the next packet sent by such wireless unit. This receive sequence number of "1" is repeated in subsequent transmissions from the wireless interface device until it successfully receives another packet, which should have a transmit sequence number of "1".

[0076] In the case in which a first host/device wireless unit sends a packet of sequence "1" which is not received by a second device/host wireless unit, the receipt by the first wireless unit of a packet of receive sequence "1" (or an ACK timeout) transmitted by the second wireless unit indicates to the first wireless unit that its transmission of the sequence "1" packet was not received and it retransmits this packet. If, on the other hand, the packet of sequence "1" transmitted by the first wireless unit is received by the second wireless unit but the acknowledgment which it transmits is lost en route to the first wireless unit, the first wireless unit will either time out the acknowledgment (and retransmit) or receive another packet, which includes an acknowledgment in the form of the correct sequence number. However, any retransmission by the first wireless unit will generally be unnecessary, since the duplicate sequence number will be detected by the second wireless unit upon receipt of the retransmitted packet and the message re-acknowledged (and the duplicate, retransmitted packet discarded).

[0077] In the event that more than three retransmissions of a packet occur, in one embodiment the communication link between the applicable host/device wireless units is reset (i.e., the send sequence number and the receive sequence number are both set to "0"). Upon receipt of a valid reset message, the receiving wireless unit acknowledges it with a packet of receive sequence number "1" and send sequence number "0". It is observed that the term "three transmissions" is intended to indicate that a set of three retransmissions of an original message have occurred, which results in the occurrence of four ACK timeouts and three BACKOFF timeouts. In one embodiment this consumes an average of approximately \((4\cdot 1.2)\cdot (3^3 (0.5+1))\) ms or 9.3 ms, depending on the device ID. This interval provides a basis for the 10 ms duration of the LISTEN timeout, which is described below.

[0078] The receipt of an ACK by a host/device wireless unit concludes a present transaction between such unit and the device/host wireless unit with which it is in communication. In the absence of any new messages to transmit, the wireless unit 112 listens for unsolicited (random access) messages from the wireless units 114, 116 following receipt of an ACK. In contrast, the wireless units 114, 116 enter a sleep mode following receipt of an ACK. In the case in which the last ACK of a transaction is lost in transit following transmission by a keyboard/mouse wireless unit 114, 116, the host wireless unit 112 retransmits the previously received message following expiration of the ACK timeout period. If the keyboard/mouse wireless unit 114, 116 enters sleep mode as soon as it transmits this "lost" ACK, it will not receive the retransmission from the host wireless unit 112 and the host wireless unit 112 will be unaware that its message has been successfully received. In view of this possibility, the keyboard/mouse wireless unit 114, 116 is configured to wait a short period in LISTEN mode prior to entering sleep mode so as to maximize the likelihood that it will receive any possible retransmission from the host wireless unit 112. As noted above, in one embodiment any message/acknowledgment transaction should complete in less than approximately 10 ms. Therefore, when a keyboard/mouse wireless unit 114, 116 sends an ACK only message, it waits until expiration of a LISTEN timeout period of 10 ms before entering sleep mode. This effectively ensures that any retransmissions from the host wireless unit 112 will be received by the keyboard/mouse wireless unit 114, 116 prior to expiration of the LISTEN timeout period.

[0079] In the exemplary embodiment, packets transmitted by the keyboard/mouse wireless unit 114, 116 are of the following general format:

1. Short and long preamble sequences (16 bits)
2. Transmit sequence number (1 bit)
3. Receive sequence number (1 bit)
4. Data present (=0)/acknowledgement only (=1) (1 bit)
5. Reserved bits set to zero (1 bit)
6. Device indication (3 bits, 0=keyboard, 1=mouse, other values reserved for future use)
7. Mouse or keyboard data (24 bits) AR zeros reserved as reset indication
8. 8 bit CRC over the entire message excluding preamble.
TABLE I
Format for packet sent from keyboard/mouse transceiver

<table>
<thead>
<tr>
<th>Header</th>
<th>Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
<td>T</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
</tr>
</tbody>
</table>

[0088]

TABLE II
Format for packet sent from keyboard/mouse transceiver (no payload case)

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

[0089]

TABLE III
Format for packet sent from host transceiver (payload case)

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

Radio Interface and Transmission Rate Selection
[0090] As was mentioned above, the radio interface 250 (FIG. 2) is representative of the set of registers and signals used in transferring messages between baseband hardware 260 within the MAC layer 208 and the remainder of each wireless unit. In the exemplary embodiment the baseband hardware 260 within the keyboard/mouse wireless units 114, 116 performs encryption/decryption, CRC generation and checking, serialization/de-serialization, and also provides an interface to the RF units 414, 514.

[0091] In the exemplary embodiment the radio interface 250 within the keyboard/mouse wireless units 114, 116 facilitates performance of at least the following functions: (i) enabling of the transmitter within the RF unit 414, 514, (ii) enabling of the receiver within the RF unit 414, 514, (iii) sending a message to the RF unit 414, 514, (iv) receiving a message from the RF unit 414, 514, (v) link quality assessment, (vi) selection of the rate used within the RF unit 414, 415, (vii) selection of the channel used within the RF unit 414, 415, (viii) control of output power of the transmitter within the RF unit 414, 415, (ix) setting the long sequence of the receiver within the RF unit 414, 415, and (x) setting the long sequence of the destination wireless unit within the transmitter of the RF unit 414, 415.

[0092] Referring to FIG. 8, an event trace diagram 800 depicts the enabling of the transmitter/receiver within the RF units 414, 514 using single-bit signals under control of the MAC layer 208. As is indicated by FIG. 8, when the wireless unit 114, 116 enters sleep mode, it disables both the transmitter and the receiver within its RF unit 414, 514.

[0093] Message Transmission
[0094] In the exemplary embodiment the MAC layer 208 initiates the sending of a message by loading the body of the message into a set of registers within the radio interface 250 and signaling “send” to the baseband hardware 260. In particular, prior to asserting “send” the MAC layer 208 loads the following information into registers within the radio interface 250:

[0095] 1. Long sequence of destination (16 bits)
[0096] 2. Header (8 bits)
[0097] 3. Pointer to message (16 bits)
[0098] 4. Power Level (2 bits)
[0099] 5. Data Rate (1 bit)
[0100] 6. Channel (3 bits)
[0101] 7. Length of message (which asserts “send”).

[0102] The “long sequence” is a 16 bit, bit sequence that is generated from the unit’s 11-bit device ID. By expanding the number of bits used to represent the ID, it is possible to distinguish between valid ID’s received without bit errors and ID that are invalid because a bit error occurred during transmission. The “header” (8 bits) comprise, transmit and receive sequence number bits, the “ACK only” bit, two zero bits, and the 3-bit device type. The “power level” is the RF
transmitter’s power level. It can be adjusted with four possible settings. Higher power for better range and reduced bit errors, lower power to conserve battery life.

[0103] Prior to initiating the process of sending a message, the MAC layer 208 calculates the proper long sequence of the destination device by writing the device ID of such device into a register within the radio interface and reading back the long sequence. In the case of the host wireless unit 112, the MAC layer 208 is configured to track both of the keyboard/mouse wireless units 114, 116 and maintain separate long sequences for each of them. The MAC layer 208 of the host wireless unit 112 also calculates a long sequence for the pairing device ID when the unit 112 is operative in a dynamic pairing (DP) mode (discussed below).

[0104] When “send” is asserted, the baseband hardware 260 continues the transmission process by sending the short sequence, which is a predetermined fixed training sequence used to bit synchronize the receiving modem to the transmitting modem, the specified long sequence, followed by the header and the body of the message itself. If the header specifies a device type of keyboard, the message portion is encrypted prior to transmission. Finally, the CRC is appended to the transmission.

[0105] Message Reception

[0106] Upon enabling of the receiver of the RF unit 414, 514, the baseband hardware 260 listens for a message with a long sequence matching the long sequence of the applicable keyboard/mouse wireless unit 114, 116. Prior to performing this enabling operation, the MAC layer 208 calculates its “own” long sequence by writing the device ID of the applicable keyboard/mouse wireless unit 114, 116 into a register within the radio interface 250. Except during pairing (discussed below), this need only be done once; during pairing, the reserved pairing device ID is instead written to the radio interface 250. The MAC layer 208 also sets a message pointer register (16 bits) in order to inform the baseband hardware 260 of the location in which the received message should be placed.

[0107] Once a matching long sequence is detected, the baseband hardware 260 receives the ensuing header and message into a header register and a message buffer (not shown), respectively. In the host wireless transceiver unit 1.12, the message is first decrypted if the header specifies a device type of keyboard. The baseband hardware 260 then checks the received CRC. If the CRC is valid, it asserts “data ready” to the MAC layer 208. Conversely, the baseband hardware 260 must not present data (i.e., assert “data ready”) to the MAC layer 208 if the CRC is invalid, although the data may already be present within the message buffer. The MAC layer 208 is configured to respond as quickly as possible to the receipt of a data message with a message in the reverse direction indicating acknowledgement.

[0108] Data Rate Selection

[0109] As was mentioned above, except where prohibited by regulation communication occurs between wireless units in the HDR mode (i.e., at 150 kb/s) by default. In order to provide more robustness in the presence of severe interference, the applicable RF units are capable of falling back to operation in the MDR mode (i.e., 50 kb/s). In the exemplary embodiment, 5 channels are available for hopping during operation in the MDR mode. In European jurisdictions, the LDR mode is employed in order to comply with the pertinent regulations. Fallback from operation in the HDR mode to the MDR mode and channel selection are controlled by the MAC layer 208. This control is facilitated by a received signal strength indication (“RSSI”) provided by the RF unit, which is asserted in the presence of strong interference within the frequency bands of interest.

[0110] Turning now to FIG. 9, there is shown a state transition diagram 900 representative of the manner in which the MAC layer 208 changes the data rate and channel of the wireless unit in which it is disposed. In the exemplary embodiment the data rate and channel of the RF unit of a wireless interface device are changed, if necessary, immediately prior to transmission of a data packet. These changes are based upon the current RSSI status and the success or failure of the previous transmission by the RF unit. When a change in channel is required, the channel sequence followed is 0-3-1-4-2-0. It has been found that adhering to a fixed sequence of this type minimizes the searching required to re-locate a partner wireless unit following a channel change. Since an interfering signal may straddle adjacent channels, it is desirable to avoid hopping to an adjacent channel; and the sequence above (and its mirror in time) is the only such sequence which prevents this from occurring.

[0111] As is indicated by FIG. 9, during operation at the medium rate the selected channel is only changed in response to link failure (which presumably occurs due to interference). Since the host wireless unit 112 rarely initiates a transaction with a keyboard/mouse wireless unit 114, 116 (i.e., most data traffic is inbound to the unit 112), while operative in medium rate mode it is possible for it to “sit” on a bad channel as it is unlikely to receive information indicating that the channel is bad. In order to avoid this situation, the host wireless unit 112 is configured to change channels every 1 to 2 seconds during operation in medium rate mode. Since each keyboard/mouse wireless unit 114, 116 is likely to be in sleep mode when this change in channel occurs, each will generally experience an ACK timeout following transmission of its next message. This timeout will also result in a change in channel. It is noted that in the case of the wireless unit 114 of the keyboard 104, the ACK timeout will cause it to reset encryption; accordingly, the host wireless unit 112 must likewise do so.

[0112] Referring again to FIG. 9, during operation of a wireless unit at the low data rate it is seen that the selected channel changes after each successful transmission. During operation at both the low and medium data rates, the channel number is reset to zero whenever the communication link is established; otherwise, whenever a transition to the medium data rate occurs, the channel last used is selected. This approach is intended to maintain channel synchronization between the wireless units at either end of the communication link. Except during operation in low data rate mode, in exemplary embodiments hardware encryption is reset whenever the channel or rate is changed; this will generally be preceded by a link reset message exchange in order to ensure timing synchronization.

[0113] Device Message Format

[0114] Referring now to Table IV below, descriptions are provided of the messages transmitted from the keyboard/mouse wireless units 114, 116 to the host wireless unit 112,
and vice-versa. As may be appreciated by reference to Table IV, each device message contains a payload of 24 bits. The bits in each message payload are utilized as follows:

[0115] Mouse report: 5 bits indicating the state of the buttons, and 6 bits for each of 3 axes containing the delta position in that axis since the last report. MS (most significant) bit is always 1.

[0116] Keyboard report: 1 bit indicating key pressed or released, 5 bits specifying the column of the keypress, and 3 bits specifying the row. The MS bit of the report is always 0. If a scroller is present, motion is reported as if the scroller were a mouse axis in a separate field. This message is always encrypted.

[0117] Reset request: This message is comprised of all zeros. The wireless keyboard does not encrypt this message, since it is sent when the keyboard and transceiver may be out of synchronization.

[0118] Pairing accept: MS bits=010, LS (least significant) bits contain device ID.

[0119] GPIO data: MS bits=011, middle byte specifies group, low byte specifies value. The section below concerning “GPIO Pins” includes additional pertinent information.

### Operational Modes and Pairing Procedure

[0120] The wireless units 112, 114 and 116 are designed for operation in either one of two modes, determined at the time of manufacturing; specifically, operation may occur in either a “dynamically paired” (DP) mode or in an “out of the box” or (OOB) mode. The DP mode is designed for business or commercial situations where multiple users may have similar devices in close proximity to each other. As is discussed below, the keyboard/mouse wireless unit 114, 116 randomly generates an ID code, which is transferred to the host wireless unit 112 at pairing time. Also at pairing time, the keyboard/mouse wireless unit 114 and 116 learn the ID code of the host wireless unit 112, which is used on all transmissions from the keyboard/mouse wireless unit 114 and 116.

[0121] The OOB mode is intended for consumer usage environments in which multiple devices are unlikely to be located closely together. This mode does not require performance of a pairing process, and thus a keyboard/mouse wireless unit 114, 116 is ready to use as soon as batteries are inserted into the wireless keyboard 104 or wireless mouse 106 in which it is disposed. In this mode, all mouse wireless units 114 use the same fixed ID code, and all keyboard wireless units 116 share a single (different) fixed ID code.

### TABLE IV

| DIRECTION | NAME | DESCRIPTION | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|-----------|------|-------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Host <>   |      |             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Link Reset |      | Reset MAC layer | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| Device Reset |      | Reset device state (keys down, etc.) | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| Shutdown Test |      | Go to sleep | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  |
| Keep Alive Test |      | Poll | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  | 3  |
| Keyboa Test |      | Enter test mode | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Board Report |      | Reset keypress/release | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| N/A |      | Future use | 0  | 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| N/A |      | Undefined | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Pairing accept |      | Scroller Up Motion (if present) | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Optical Mouse |      | Register # | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| GPIO |      | Register # | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Config-Set Mouse Report |      | Register # | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Mouse Report |      | Register # | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

Format (MSb <-> LSb, 24 bits)
As may be appreciated from the above, each wireless unit 112, 114, 116 is either pre-assigned a device ID or are assigned a dynamically generated device ID at pairing. With the exception of a few reserved sequences (e.g., all "0" or all "1"), in the exemplary embodiment all 11-bit device IDs are valid. If a device is in DP mode, at pairing time it generates a random 11-bit ID in hardware upon powering-up. This device 112 is passed through a hardware block described in the above-referenced copending provisional patent application, which converts the device ID into a 16-bit long sequence which is incorporated within messages intended for receipt by the applicable wireless unit. Prior to enabling the receiver within its RF unit, a wireless unit 112, 114 and 116 writes its 16-bit long sequence to its radio interface. During operation, the receiver listens for messages having addresses containing this long sequence. Whenever a message is transmitted a wireless unit 112, 114, 116, a long sequence prefix identifies the ID code of the intended recipient. In the DP mode, the host wireless transceiver unit 112 does not possess a priori knowledge of the device ID of the keyboard/mouse wireless unit 114, 116. Rather, during operation in DP mode a "pairing" operation is performed pursuant to which a device ID is dynamically generated by the keyboard/mouse wireless unit 114, 116 and communicated to the host wireless unit 112. This operation is performed following each loss of power within the keyboard/mouse wireless unit 114, 116 (e.g., whenever the batteries of a unit 114, 116 are removed).

In one embodiment of the OOB mode, each transceiver is identified by a predefined address. Specifically, a host transceiver is assigned TR-DEFAULT as an address, a keyboard transceiver is assigned KB-DEFAULT as an address, and a mouse is assigned M-DEFAULT as an address. Following completion of a successful pairing operation between a host and device transceiver, the paired transceivers will have unique addresses. In addition, in this embodiment a host transceiver assumes its unique address when communicating with a paired device transceiver. However, the host transceiver will continue to use TR-DEFAULT during communication with unpaired device transceivers.

In the case in which EEROM is included within the host/device transceivers, in one embodiment all will have unique addresses. In the case in which the host/device transceivers lack an EEPROM, in one embodiment the host transceiver uses the preamble corresponding to the TR-DEFAULT address. In the case in which the host/device transceivers include an EEPROM, the host transceiver uses the preamble corresponding to its unique address.

FIG. 12 is an event trace diagram 1200 representative of the messages exchanged during a pairing operation involving a keyboard/mouse wireless unit 114, 116 (a "device") and the host wireless unit 112. As shown, the pairing operation is initiated by the manual push of a predefined button of the host wireless unit 112 followed by the pressing of a predefined button of the keyboard 104 or mouse 106 associated with the device within a few seconds. When the host wireless unit 112 is incorporated within a dongle (not shown) attached to the host PC 102, a dedicated button may be provided on the dongle for use during pairing. Success of the pairing operation is evidenced by proper operation of the device (e.g., by keystrokes or mouse clicks being processed by the host PC following receipt by the host wireless unit 112). If the pairing operation fails, both sides continue to attempt to pair until success or expiration of a predefined number of attempts (e.g. 1000). If the timeout period expires, the device goes to sleep until the pairing button of the keyboard 104 or mouse 106 associated with the device is pushed again.

As is indicated by FIG. 12, when the predefined button activating pairing operation of the host wireless unit 112 is pressed, the host wireless unit 112 begins transmitting "pairing request" messages which contain the transceiver ID code of the host wireless unit 112 and which are addressed to a reserved ID code (e.g., all "1"). When the predefined button of the keyboard 104 or mouse 106 attached to the device is pressed, the device temporarily assumes the reserved ID and listens for pairing requests from the host wireless unit 112. If the device successfully receives a request, it responds with an ACK containing a pairing accept message to the transceiver ID code. This message contains the actual device ID code of the transmitting device.

The host wireless unit 112, between issuing pairing request messages, listens for incoming messages. If it receives a pairing accept message from the device, it remembers the ID code of its new partner and acknowledges it. Once, the acknowledgement is received at the device, the device assumes its "rear" identity; that is, it begins sending and receiving messages on the basis of its actual device ID. Once the device has assumed its actual identity, it transmits a reset message to the host wireless unit 112. If the reset message is acknowledged by the host wireless unit 112, the applicable communication link 108, 110 is considered established and normal operation ensues.

When the pairing button associated with the host transceiver is pressed, in the exemplary embodiment the host transceiver transmits pairing request messages periodically (every 50 ms in the case of HDR), using a fixed PAIRING-PREAMBLE. The pairing request message will generally contain the unique address of the host transceiver (32 bits), and host transceiver preamble (16 bits). When the pairing button triggering the keyboard/mouse transceiver is pressed, it will listen to pairing request messages for a pairing interval (55 ms in the case of HDR) using the fixed PAIRING-PREAMBLE. When this period expires, the keyboard/mouse transceiver checks whether there are any user input events. If there are user events, the keyboard/mouse transceiver transmits them for (2-period–pairing interval) (2×50–55=45 ms for HDR). If there are not any user inputs, the keyboard/mouse transceiver immediately returns to the pairing process. If the keyboard/mouse transceiver successfully receives a pairing request, it responds with an ACK containing a pairing accept message using the preamble of the host transceiver. This message contains the unique address (32 bits) of the keyboard/mouse transceiver. If the host transceiver hears the pairing accept message from the keyboard/mouse transceiver, it remembers the address of the keyboard/mouse transceiver and acknowledges the pairing accept message. When this acknowledgement is received at the keyboard/mouse transceiver, the keyboard/mouse transceiver assumes the its new address.

Device Configurability Options

Referring now to FIG. 11, there is shown a computer system 1100 in accordance with the present invention to which reference will be made in describing one manner in
which various parameters of wireless units operative within the system 1100 may be configured upon establishment of radio communication links between such units. As shown, the system 1100 includes a host wireless unit 1112 of a host PC 1102, a wireless unit 1114 of a wireless keyboard 1104, and a wireless unit 1116 of a wireless mouse 1106.

[0130] In the embodiment of FIG. 11, costs of implementing the system 1100 are minimized by designing the host wireless unit 1112 to incorporate an EEPROM 1134 as well as a host transceiver module 1130. In this regard the host transceiver module 1130 is disposed to implement all of the above-described functionality of the host wireless unit 112. In addition, the EEPROM 1134 stores configuration options for each of the wireless units of the system 1100; that is, configuration information is stored for the host transceiver module 1130 and keyboard/mouse wireless units 1114, 1116. By taking advantage of the two-way radio links between the host transceiver module 1130 and keyboard/mouse wireless units 1114, 1116, the parameters stored within the EEPROM 1134 may be distributed to the keyboard/mouse wireless units 1114, 1116 upon the establishment of radio communication within the system 1100.

[0131] In certain embodiments the keyboard/mouse wireless units 1114, 1116 each include a small amount of writable non-volatile memory (e.g., 4KB). Alternatively, the keyboard/mouse wireless units 1114, 1116 do not include writable non-volatile memory, and power-up in a fixed state defined only by hardware inputs and built-in defaults. Since requiring multiple pins for configuration is cumbersome and expensive, in the exemplary embodiment each keyboard/mouse wireless unit 1114, 1116 uses only one pin for configuration input to specify operation in the DP or OOB mode. Upon power-up of a keyboard/mouse wireless unit 1114, 1116, various operational parameters are set to the default values stored within corresponding locations of its writable non-volatile memory or registers.

[0132] Once a radio link has been established between the host transceiver module 1130 and a 30 keyboard/mouse wireless unit 1114, 1116, the host transceiver module 1130 reads the EEPROM 1134 and identifies those operational parameters of the keyboard/mouse wireless units 1114, 1116 with respect to which the default values should be overridden. For each of these identified parameters, the host transceiver module 1130 sends a configuration message to the keyboard/mouse wireless unit 1114, 1116. Each configuration message includes a parameter ID uniquely identifying a register or the location within the non-volatile memory of the keyboard/mouse wireless unit 1114, 1116, as well as a parameter value to be stored in such location. As mentioned above, the value of each parameter is initialized upon power-up of the keyboard/mouse wireless unit 1114, 1116 to a default value; however, the sending of a configuration message permits the default value to be overridden with a desired value appropriate for the intended application and/or system environment. In the exemplary embodiment the keyboard/mouse wireless unit 1114, 1116 does not attempt to validate or otherwise check these values; rather, it simply writes the selected register or memory location and proceeds.

[0133] In summary, for each parameter to be changed in a keyboard/mouse wireless unit 1114, 1116, the host wireless unit 1112 issues a configuration message containing the parameter ID and the desired value. These values simply replace the defaults until the keyboard/mouse wireless unit 1114, 1116 loses power. Upon again becoming energized, the ‘keyboard/mouse wireless unit 1114, 1116 re-establishes its radio link with the host wireless unit 1112, and the parameter overrides are retransmitted.

[0134] If no parameter overrides are required and the system 1100 is operating in OOB mode, then no EEPROM 1134 is required within the host the host wireless unit 1112. However, the absence of the EEPROM 1134 is generally not a sufficient basis upon which to assume that operation in the OOB mode is intended, as the apparent absence of EEPROM 1134 may in fact be the result of a failure. In order to detect this situation, one configuration input is used at the hardware pin level. Specifically, pulling up this pin with a resistor 1140 indicates that EEPROM 1134 is not present.

[0135] As the host wireless unit 1112 and the keyboard/mouse wireless unit 1114, 1116 each have built-in defaults for all parameters, only variations need be stored within the EEPROM 1134. Accordingly, in implementations of the system 1100 in which no changes are required to be made to these built-in defaults, no EEPROM 1134 is required. That being said, if the system 301100 is intended to be used in DP mode, it will necessary for some provision to be made for the host wireless unit 1112 to remember its pairings when the host PC 1102 is turned off and the host wireless unit 1112 loses power.

[0136] In the embodiment of FIG. 11, it is necessary for each keyboard/mouse wireless unit 1114, 1116 to know in which mode (DP or OOB) it is to be operative upon powering up. Since in this case only two possible operational modes exist, only one bit of configuration information need be provided in order to select between modes. By providing the option of adding a mode selection resistor 1150, 1152 to each the keyboard/mouse wireless units 1114, 1116, providing this configuration information to each device may be achieved at a “cost” of only one resistor per device and obviates the need to provide additional pins. In an exemplary embodiment the presence of the resistor 1150, 1152 results in the keyboard/mouse wireless unit 1114, 1116 becoming operative in the OOB mode (i.e., the device uses a single static device ID for communicating with the host wireless unit 1112. If the resistor 1150, 1152 is absent, then the keyboard/mouse wireless unit 1114, 1116 elects a random device ID and expects the pairing procedure discussed above to be followed.

[0137] In order to provide positive confirmation of the validity of the contents of the EEPROM 1134, the records for each wireless unit 1112, 1113 are protected by checksums which are verified each time the host wireless unit 1112 powers up. As discussed above, in order to resolve the ambiguity, which may exist if the EEPROM 1134 is not accessible upon the host wireless unit 1112 powering on (i.e., either the EEPROM 1134 is present but defective or is simply not present), the resistor 1140 is used to indicate whether or not the EEPROM 1134 should be present. If the resistor 1140 is present, no attempt is made to access an EEPROM 1140 and the system 1100 operates utilizing its default values. If the resistor 1140 is absent, then the EEPROM 1134 must be present and checksum tests are performed upon the records of the EEPROM 1134.
Referring now to Table V, a list of exemplary default values and ranges is provided for various operational parameters (or, equivalently, “options”) of the keyboard wireless unit 1114. For each operational parameter that is to be set at something other than its default value, a corresponding option value is transferred from the host wireless unit 1112 to the keyboard wireless unit 1114 once a communication link has been established between the units 1112, 1114. It is observed that a set of keyboard-related options are also implemented by the host wireless unit 1112 (see Table IV below), and thus need be transferred to the keyboard wireless unit 1114.

### TABLE V

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>K1</td>
<td>Paired/Out of Box (OOB) Mode</td>
<td>OOB</td>
<td>Paired</td>
<td>OOB</td>
</tr>
<tr>
<td>K3</td>
<td>Scan Rate (# of scans for debounce)</td>
<td>5</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>K4</td>
<td>Scan Time (ms between scans)</td>
<td>4</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>K6</td>
<td>“Same Keys Down” Timeout (seconds)</td>
<td>2</td>
<td>0 (never)</td>
<td>63</td>
</tr>
<tr>
<td>K7</td>
<td>Phantom Timeout (seconds)</td>
<td>20</td>
<td>0 (never)</td>
<td>63</td>
</tr>
<tr>
<td>K8</td>
<td>Keep-Alive (Out of Range Detection) Interval (seconds)</td>
<td>0 (Do not wake up)</td>
<td>0 (Do not wake up)</td>
<td>255</td>
</tr>
</tbody>
</table>

[0139] Table VI provides a list of exemplary default values and ranges for various options of the mouse wireless unit 1116. For each operational parameter that is to be set at something other than its default value, a corresponding option value is transferred from the host wireless unit 1112 to the mouse wireless unit 1116 once a communication link has been established between the units 1112, 1116. A large percentage of an optical mouse’s battery life is typically consumed by the mouse’s optical sensor. To conserve battery life, the sensor operation is suspended periodically after movement of the mouse is stopped. The suspension duration is increased in a stepwise fashion as the time since the last movement is increases. In the preferred embodiment there are four steps, S0, S1, S3 and S4. The duration of each step and the and the duration of the suspension during each step is configurable.

[0140] It is observed that a set of mouse-related options are also implemented by the host wireless unit 1112 (see Table VII below), and thus need be transferred to the mouse wireless unit 1114.

### TABLE VI

<table>
<thead>
<tr>
<th>Option #</th>
<th>Description</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>Paired/Out of Box (OOB) Mode</td>
<td>OOB</td>
<td>Paired</td>
<td>OOB</td>
</tr>
<tr>
<td>M3</td>
<td>Button Scan Rate [# of scans for debounce]</td>
<td>5</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>M4</td>
<td>Button Scan Time [ms between scans]</td>
<td>4</td>
<td>1</td>
<td>7</td>
</tr>
</tbody>
</table>

### TABLE VI-continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>M5</td>
<td>Mouse Type</td>
<td>Mech.</td>
<td>Mech.</td>
<td>Optical</td>
</tr>
<tr>
<td>M6</td>
<td>S0 → S1 Timeout [ms]</td>
<td>0</td>
<td>200</td>
<td>255</td>
</tr>
<tr>
<td>M7</td>
<td>S1 Sample Rate [ms]</td>
<td>10</td>
<td>1</td>
<td>255</td>
</tr>
<tr>
<td>M8</td>
<td>S1 → S2 Timeout [sec]</td>
<td>60</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>M9</td>
<td>S2 Sample Rate [ms]</td>
<td>65</td>
<td>1</td>
<td>255</td>
</tr>
<tr>
<td>M10</td>
<td>S2 → S3 Timeout [10 sec]</td>
<td>120</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>M11</td>
<td>S3 Sample Rate [ms]</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>M12</td>
<td>Keep-Alive (Out of Range Detection) Interval (seconds)</td>
<td>0 (Do not wake up)</td>
<td>0 (Do not wake up)</td>
<td>255</td>
</tr>
</tbody>
</table>

### TABLE VII

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TK1</td>
<td>Keyboard Present?</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TK2</td>
<td>Keyboard Device ID (high byte)</td>
<td>OOB value</td>
<td>(H)</td>
<td></td>
</tr>
<tr>
<td>TK3</td>
<td>Keyboard Device ID (low byte)</td>
<td>OOB value</td>
<td>(L)</td>
<td></td>
</tr>
<tr>
<td>TK4</td>
<td>Battery Levels to Report (no report)</td>
<td>2</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>TK5</td>
<td>Use Alternate Scan Code Table?</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TK6</td>
<td>Scroller Type</td>
<td>Volume Control</td>
<td>Volume Control</td>
<td>Scroll Keys</td>
</tr>
<tr>
<td>TK100-</td>
<td>Alternate Scan Code Table</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>TK260-</td>
<td>Group 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM1</td>
<td>Mouse Present?</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TM2</td>
<td>Mouse Device ID (high byte)</td>
<td>OOB value</td>
<td>(H)</td>
<td></td>
</tr>
<tr>
<td>TM3</td>
<td>Mouse Device ID (low byte)</td>
<td>OOB value</td>
<td>(L)</td>
<td></td>
</tr>
<tr>
<td>TM4</td>
<td>Optical Mouse Frame Rate (high limit) (high byte)</td>
<td>0</td>
<td>1500 (H)</td>
<td>255</td>
</tr>
<tr>
<td>TM5</td>
<td>Optical Mouse Frame Rate (low limit) (low byte)</td>
<td>1500 (L)</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>TM6</td>
<td>Optical Mouse Frame Rate (high limit) (low byte)</td>
<td>0</td>
<td>1500 (H)</td>
<td>255</td>
</tr>
</tbody>
</table>

[0141] Table VII provides a list of exemplary default values and ranges for various options of the host transceiver module 1130. In the exemplary embodiment there exist three logical groups of options processed by the host transceiver module 1130. The first and second option groups (i.e., Group 1 and Group 2) relate to the operation of the keyboard/mouse wireless units 1114, 1116, but are processed locally by the host transceiver module 1130. The third logical group (i.e., Group 3) of Table VII consists of options pertinent to operation of the host transceiver module 1130 itself.
TABLE VII-continued

<table>
<thead>
<tr>
<th>Option #</th>
<th>Description [unit]</th>
<th>Default Value</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM7</td>
<td>Optical Mouse Frame Rate (low limit) (low byte)</td>
<td>1500 (L) (&lt;200)</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>TM8</td>
<td># of Mouse Buttons</td>
<td>3</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>TM9</td>
<td>Inches (multiple of 10 cpi; 20 = 200 cpi)</td>
<td>20?</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>TM10</td>
<td>Battery Levels to Report</td>
<td>2</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>TM11</td>
<td>PS2 Commands光学 Mouse INIT Command Count</td>
<td>Standard Standard Standard BTC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TM100</td>
<td>Optical Mouse INIT Command Count</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>TM101</td>
<td>Init Command 1 Register</td>
<td>—</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>TM102</td>
<td>Init Command 1 Value</td>
<td>—</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>TM103 up</td>
<td>Rest of Init Commands (2 options per command, up to a maximum of TM130)</td>
<td>—</td>
<td>0</td>
<td>255</td>
</tr>
</tbody>
</table>

Group 3

| TT1      | Transceiver Device ID (high byte) OOB value (H) | — |
| TT2      | Transceiver Device ID (low byte) OOB value (L) | — |
| TT3      | European Operation (low rate hopping) No No Yes |
| TT4      | Encryption Key | — |

Device Customization and Application Development

In addition to enabling the wireless units 114, 116 to be flexibly configured in the manner described in the previous section, in one aspect the present invention further contemplates that the operation of the wireless units 114, 116 be amenable to customization in view of the requirements of various applications. To this end, the software executing on the wireless units 112, 114, 116 has been designed to incorporate various “hooks” in order to allow certain parameters adjusted. Such hooks are also intended to allow portions of the software to be extended or replaced by supplementary software included within an inexpensive external serial “patch” EEPROM of the general type described above with reference to FIG. 11.

Referring to FIG. 22, a high-level representation is provided of a system 2200 comprising of wireless units incorporating EEPROM patches in accordance with one aspect of the present invention. In particular, the system 2200 includes a host wireless unit 2212 incorporating transceiver software within a ROM 2222 and transceiver patch software within an EEPROM 2232. The system 2200 further includes a keyboard wireless transceiver 2214 containing a ROM 2224 in which is stored keyboard software and an EEPROM 2234 in which is stored keyboard patch software. Finally, the system 2200 also includes a mouse wireless transceiver 2216 configured with a ROM 2224 incorporating mouse software and an EEPROM 2236 incorporating mouse patch software.

In general, the host transceiver 2212 and device transceivers 2214, 2216 are ROM-based designs. That is, the bulk of the program code stored within these transceivers is normally executed from ROM. However, the inclusion of serial EEPROM within one or all of the transceivers 2212, 2214 and 2216 facilitates a patching procedure allowing certain default operational parameters to be changed at startup and limited changes to be made to the operating software of the transceiver.

FIG. 23 is a flowchart representative of certain aspects of the operation of the system 2200. As shown in FIG. 23, upon powering-up the processor (not shown) within each transceiver 2212, 2214, 2216 initializes critical hardware and software then checks the Fourier bus to which it is connected for the presence of an external EEPROM. If an EEPROM is present, the processor reads the first few EEPROM bytes and compares them to a predefined number sequence. If the sequence matches, the content of the EEPROM is copied to specified locations in the processor's external RAM. Execution of the software stored within the ROM 2222, 2224, 2226 of the transceiver 2212, 2214, 2216 then continues.

Within a “config” data block in the external RAM of a processor included within each host/device transceiver 2212, 2214, 2216, the software stored within ROM 2222, 2224, 2226 expects to find certain critical operating parameters. This data block is initialized by the software stored within ROM 2222, 2224, 2226 upon start up. If no EEPROM patch software is present, the software stored within ROM 2222, 2224, 2226 uses the values loaded at start up. If a patch EEPROM 2232, 2234, 2236 is present, the config data block can optionally be over-written as the contents of the patch EEPROM 2232, 2234, 2236 are read.

[0142] GPIO Pins

[0143] In order to enhance the potential for configurability of the wireless units 114, 116, each may include a number of external pins defined as general purpose I/O (GPIO). Electrically, each of these pins will generally be implemented as an open drain output with the ability to be read back as an input if an associated output register contains a “1” bit. Upon power-up, the device wireless unit 1114, 1116 unit initializes its GPIO pins to “1” values and thereafter is not involved in changing their values. For reference purposes, GPIO pins are grouped into sets of eight, which are manipulated in parallel. Each group of eight is assigned a group number.

[0144] Under certain circumstances the host wireless unit 112 may request to set or read a group of GPIO pins of a wireless unit 1114, 1116. To this end the host wireless unit 1112 conveys a request to the device wireless unit 1114, 1116 by way of a GPIO read or GPIO write message containing an identification of the group number. In the case of a GPIO write message, the value to be written is also provided. Upon receipt of a GPIO read message, the device wireless unit 1114, 1116 responds with a GPIO data message containing the requested data. It is the responsibility of the host wireless unit 1112 to write a “1” to any input bit before it is read.
The software stored within ROM 2222, 2224, 2226 also expects to find a control table potentially containing pointers to code patches within external RAM of the applicable processor. Each pointer potentially points to a ROM code replacement function. For most entries in this table a null pointer value is expected. If a null entry is found, the software stored within ROM 2222, 2224, 2226 executes the original ROM code function for that pointer location. However, if the software within the patch EEPROM 2232, 2234, 2236 over-writes the control table value with a non-null pointer, the replacement function referenced by the pointer is executed in place of the ROM code function. Replacement functions can be other functions previously defined in ROM 2222, 2224, 2226, or newly-defined functions loaded from the patch EEPROM 2232, 2234, 2236 to RAM at startup.

Consistent with the invention, the EEPROM patching procedure discussed above may be used for a variety of purposes including, for example, for default parameter substitutions, procedure substitutions, and the execution of precompiled functions. The first two of these potential uses is summarized below.

Parameter Substitutions

Those parameters assigned default ROM values which may be replaced (i.e., “substitutable parameters”) are defined in a predefined file within the patch EEPROM 2232, 2234, 2236. This file declares a predefined type (i.e., “Mask ROM Config”) which, when instantiated, defines the available substitutable parameters.

Procedure Substitutions

A predefined file (i.e., “MRCONTH”) contains the type definition for the Mask ROM Control “t” type referenced above. By default this structure is instantiated as a series of null function pointers. Each null pointer is a placeholder for a potential substitute procedure of the general form illustratively represented by FIG. 24. In order to substitute a new function for a default function, the null function pointer corresponding to the default procedure is replaced with a function pointer pointing to the new replacement procedure. Replacement procedures are located in the RAM program area (loaded from the applicable patch EEPROM 2232, 2234, 2236 on power up). Alternatively, replacement procedures can reference the preexisting functions in the code within ROM 2222, 2224, 2226.

Operation of Host Wireless Unit

As may be appreciated by reference to FIG. 3-5, in the exemplary embodiment the architecture of the host wireless unit 112 (or, equivalently, “host transceiver 112”) is substantially similar to that of the device wireless units 114, 116. Accordingly, for the sake of clarity of presentation the following discussion does not repeat those aspects of the preceding discussion equally applicable to the host and device wireless units, and is instead intended to highlight the structural and operational differences there between.

Host Transceiver Message Format

As mentioned above, Table 1 provides information regarding each of the messages transmitted from the keyboard/mouse wireless units 114, 116 to the host wireless unit 112, and vice-versa. As may be appreciated by reference to Table 1, the host transceiver 112 sends messages containing the following 24-bit payloads:

Pairing request: Most significant (MS) byte identifies message, least significant (LS) bits specify device ID. Only sent to reserved pairing ID.

Configuration: MS byte identifies message, middle byte specifies a configuration register or memory location, and low byte contains data to be written to the specified register or memory location.

Optical mouse command: MS byte identifies message, middle byte specifies optical mouse chip register, low byte specifies value to be written to the optical mouse chip.

GPIO read: MS byte identifies command, middle byte specifies a GPIO group, LS byte unused.

GPIO write: MS byte identifies command, middle byte specifies a GPIO group, LS byte contains data to be written to the GPIO pins of the specified group.

Shutdown message: Sent only on behalf of the host PC 102. The destination wireless unit 114, 116 sends an ACK upon receipt of a Shutdown message, then goes to its lowest-power state. Note that a wake-up message, which is described in the above-referenced copending patent application, will not “rouse” a wireless unit 114, 116 from its lowest-power state.

Keep-Alive: Optionally used in order to detect when a wireless unit 114, 116 moves out of range of the host wireless unit. In response to receipt of a Keep-Alive message, the wireless unit 114, 116 sends an ACK and then may sleep for the configured keep-alive interval.

Power Management

In the exemplary embodiment the system 100 implements a power management scheme consistent with the “OnNow” initiative for system-wide power management defined by Microsoft Corporation. As part of this initiative, Microsoft has published “Device Class Power Management Reference Specifications” for various device classes, including the input device class applicable to keyboards, mice, joysticks, and the like (see, e.g., www.microsoft.com). The Reference Specifications for the input device class defines four power states, DO through D3. A device operative in the DO state is completely active and normally functioning, and consumes power at the highest level possible. In contrast, state D3 corresponds to the case in which power may have been fully removed from the device, and it considered to be completely “off”. As state D2 has been characterized as being inapplicable to input devices, the only remaining state is D1. The D1 state requires that device power consumption be equal to or greater than the D2 state, but less than the DO state. Accordingly, in the D1 state hardware such as displays and indicators (e.g., LED devices) are turned off, but state information such as num, caps, and scroll lock state are preserved. In the exemplary embodiment transitions between power states within a device are explicitly commanded by drivers within the PC host, and are not performed autonomously.
Consistent with one aspect of the present invention, host wireless unit 112 and keyboard/mouse wireless units 114, 116 define a power down state D1 as follows:

Upon receipt of a “go to D1” command via a PS/2 port, the host wireless unit 112 transmits a power down message to all connected keyboard/mouse wireless units 114, 116.

The host wireless unit 112, after receiving all necessary ACK’s, shuts off its RF unit 314 (both transmitter and receiver) and all LED devices.

After performing the above shut down operations, the processor of the host wireless unit 112 halts in a mode where a PS/2 interrupt or power cycling is the only way to wake it up. The wireless unit 112 should be in the lowest power consumption state possible, consistent with receiving a PS/2 message.

A keyboard/mouse wireless unit 114, 116, upon receiving a power down message, sends an ACK in response and waits for an ACK timeout period in order to ensure that the ACK is received. Following expiration of this time out period, the wireless unit 114, 116 shuts down both the transmitter and receiver of its RF unit 414, 514. In the case of the mouse wireless unit 116, instructions are also provided to any optical mouse chip within its wireless mouse 106 to enter a power down state.

The keyboard/mouse wireless unit 114, 116 then enters a sleep mode until a key of the wireless keyboard 104 or wireless mouse 106 is pressed, as applicable.

Once the keyboard/mouse wireless unit 114, 116 wakes up in response to the occurrence of such a key press, it resumes operation, retaining the device ID of any paired devices. Such resumption of operation generally entails: (i) restoring the LED devices of the wireless keyboard/mouse 104, 106, and (ii) turning on the receiver of its RF unit 414, 514, utilizing the same rate and channel selections as were in place prior to sleep mode being entered.

Upon waking up, the keyboard/mouse wireless unit 114, 116 also attempts to send a reset message to the host wireless unit 112. Failure is handled as before; that is, the key press initiating the wakeup operation is discarded.

Collision Avoidance and Interference Handling

Collision Avoidance

FIG. 14 depicts a flowchart 1400 representative of a carrier sense/clear channel assessment procedure (CS/CCA) procedure executed by the MAC layer 208 of the keyboard/mouse wireless unit 114, 116 prior to transmission of a message. As shown, in the event a message needs to be transmitted by the keyboard/mouse wireless unit 114, 116, the receiver of its RF unit 414, 514 is enabled and it waits for a predefined period for PLL locking to occur. Next, the receiver of its RF unit 414, 514 provides a CCA report indicative of the level of interference present within the channel in which it desired to transmit the message. In the exemplary embodiment a 1-bit CCA report is provided by the RF unit 414, 514; namely, a CCA of “1” indicates the channel is busy, and a CCA of “0” indicates the channel is clear. The details of the remainder of CS/CCA procedure is illustrated by FIG. 14.

The duration of the random backoff (B) occurring within the CS/CCA procedure FIG. 14 may be expressed as

\[
B = B_{\text{min}} + N * W
\]

where:

\[
B_{\text{min}} = \text{Packet duration} + T_{\text{tx}} + T_{\text{rx}} + T_{\text{MAC}} + T_{\text{PLL,ACK}}
\]

\[
T_{\text{MAC}} = \text{MAC processing time (assuming 150 \mu s)}
\]

\[
T_{\text{PLL,MAC}} = \text{PLL switching time=150 \mu s}
\]

\[
T_{\text{tx}} = \text{transmitter latency}
\]

\[
T_{\text{rx}} = \text{receiver latency}
\]

and where \(W = 0.5 * B_{\text{min}} \) and \(N \) is a random integer (e.g., \(0 \leq N \leq 3\)). In addition, the “window 1” referenced in FIG. 14 corresponds to the on-air turnaround time between the transmitter and receiver of different wireless units in the case of an ACK, and is given by

\[
T = T_{\text{PLL}} + T_{\text{MAC}} + T_{\text{tx}}
\]

The transmitter latency (Ttx) and receiver latency (Ttx) for various data rates are provided in Table V, and the durations of various types of message packets are set forth in Table VI.

<table>
<thead>
<tr>
<th>Data rate</th>
<th>Ttx (\mu s)</th>
<th>Ttx (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>112.5 kb/s</td>
<td>29</td>
<td>21</td>
</tr>
<tr>
<td>28.125 kb/s</td>
<td>82</td>
<td>67</td>
</tr>
<tr>
<td>9.375 kb/s</td>
<td>224</td>
<td>138</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data rate</th>
<th>Packet duration for keyboard (\mu s)</th>
<th>Packet duration for mouse (\mu s)</th>
<th>Duration for “piggy-back” ACK (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>112.5 kb/s</td>
<td>596</td>
<td>738</td>
<td>312</td>
</tr>
<tr>
<td>28.125 kb/s</td>
<td>2383</td>
<td>2952</td>
<td>1245</td>
</tr>
<tr>
<td>9.375 kb/s</td>
<td>7147</td>
<td>8854</td>
<td>3734</td>
</tr>
<tr>
<td>Spreading</td>
<td>6556</td>
<td>8854</td>
<td>3452</td>
</tr>
</tbody>
</table>

Table VII provides a tabular listing of the value of Bmin as a function of data rate.

<table>
<thead>
<tr>
<th>Data rate</th>
<th>Bmin (\mu s)</th>
<th>Bmin (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>112.5 kb/s</td>
<td>1.287</td>
<td>1.429</td>
</tr>
<tr>
<td>28.125 kb/s</td>
<td>4.159</td>
<td>4.728</td>
</tr>
<tr>
<td>9.375 kb/s</td>
<td>11.767</td>
<td>13.474</td>
</tr>
<tr>
<td>Spreading</td>
<td>10.367</td>
<td>11.929</td>
</tr>
</tbody>
</table>

Collision Avoidance and Interference Handling
In the exemplary embodiment the CCA threshold level utilized by the RF unit 414, 514 in generating the CCA report during execution of the CS/CCA procedure illustrated by FIG. 14 does not remain the same for all applications. Rather, even for a given application, the CCA 20 threshold level will generally depend upon the distance between the transmitting/receiving devices as well as the distance between an interferer and such devices.

Turning now to FIG. 15, a flowchart 1500 is provided of procedure for dynamically adjusting the CCA threshold level used by the RF unit 414, 514 when generating a CCA report. In the embodiment of FIG. 15, the default values are: \( M = -60 \text{dBm}, N = 2000 \text{packets}, x = 60\%, y = 10\%, a = 2\%, b = 7\% \), where \( M, N, x, y, a, \) and \( b \) are configurable parameters.

Table VIII shows an exemplary mapping between CCA threshold level (TH) and RF control signals:

<table>
<thead>
<tr>
<th>CCA_Vth_b2</th>
<th>CCA_Vth_b1</th>
<th>CCA_Vth_b0</th>
<th>CCA threshold level (TH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-48 dBm</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-54 dBm</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-60 dBm</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-66 dBm</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-72 dBm</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-78 dBm</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-84 dBm</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-90 dBm</td>
</tr>
</tbody>
</table>

In table VIII, it is noted that (CCA_Vth_b2=1, CCA_Vth_b1=0, CCA_Vth_b0=0) and (CCA_Vth_b2=1, CCA_Vth_b1=0, CCA_Vth_b0=1) are not used. CCA_Vth_b2, CCA_Vth_b1 and CCA_Vth_b0 are three control line bits used to set the CCA threshold level.

**Interference Handling**

FIG. 16 is a high-level flow diagram representative of a general approach to interference handling consistent with the present invention. As was discussed above, data is transmitted and received by wireless devices in four different modes: a high data rate (HDR) mode; a medium data rate (MDR) mode; a low data rate (LDR) mode and a spreading mode. The HDR mode is the default mode, which can provide 150 kbps data transmission. The data rates for the MDR, LDR and spread mode are 30 kbps, 10 kbps and 13.64 kbps respectively. Spreading mode is used when there is interference from a non-mobile device(s) (e.g., other host and device transceivers), and MDR is used when there is interference from a non-mobile device(s) (e.g., narrow-band interference) such as from citizen band (CB) radio. In certain embodiments each wireless unit 112, 114, 116 is able to detect interference and switch to appropriate modes automatically. As a consequence, highly reliable wireless data transfers may be affected even in an environment with multiple other wireless users.

**Normal/Spreading Mode and Spreading/Normal Mode Switching**

Consistent with one aspect of the invention, spreading mode may be invoked to mitigate interference engendered by the presence of other host/device transceivers within the vicinity of a host/device transceiver pair desiring to communicate. In an exemplary embodiment the device transceiver of a host/device transceiver pair controls the switching between normal and spreading mode.

A device transceiver 114, 116 initiates spreading mode operation by sending, to the host transceiver 112, a packet, which has been spread in the manner described in the above-referenced copending patent application. Upon receiving a packet in spreading mode, the modem 308 at the host transceiver 308 will notify the MAC layer 208 that spreading mode has been enabled, which results in the host transceiver 112 also sending the ACK in spreading mode.

If the host transceiver 112 and device transceiver 114, 116 are in spreading mode and the device transceiver 114, 116 makes decision to switch to normal mode, the device transceiver 114, 116 will send a packet in normal mode. Upon receiving this packet in normal mode, the modem 308 at the host transceiver 112 notifies the MAC layer 208 that the device transceiver 114, 116 has switched back to normal mode. This results in the host transceiver 112 sending the ACK in normal mode.

Of course, a given host transceiver 112 will generally be in communication with more than one device transceiver 114, 116. Accordingly, the host transceiver 112 is disposed to 20 communicate with each device transceiver 114, 116 in accordance with the mode that has been currently selected by such device transceiver 114, 116.

In the exemplary embodiment the device transceivers 114, 116 elect to switch from normal mode to spreading mode when more than a predefined number of failures occur within a predefined number of the most recent transmissions. For example, it has been found that switching to spreading mode is appropriate if more than 21 failures occur in connection with the most recent 25 attempted transmissions. Similarly, the device transceivers 114, 116 may elect to switch from spreading mode to normal mode when less than a predefined number of failures occur within a predefined number of the most recent transmissions. For example, it has been found that a return to normal mode from spreading mode may be when less than 18 failures occur in connection with the most recent 25 transmissions.

**Switching Between HDR and MDR**

FIG. 17, a flow chart 1700 is provided of an exemplary manner in which the host transceiver 112 transitions between operation within the HDR and MDR modes, and vice-versa. In the exemplary embodiment a transition is made from HDR to MDR mode when a received signal strength indication (RSSI) generated by the RF unit 314 indicates the presence of strong interference (e.g., narrow-band interference) such as from citizen band (CB) radio.

Within the host transceiver 112, the RSSI generated by the RF unit 314 is checked periodically (e.g., every 2 ms). If it is determined to be necessary to switch to MDR mode, switching occurs to a particular MDR channel in accordance with the sequence 0-3-1-4-2-0... In the case of the initial switch from HDR to MDR mode, the host transceiver 112 first attempts to operate in a predefined MDR channel (e.g., channel 4). If in fact "L" is the channel through which the host/device transceivers are communicating immediately
prior to transitioning back to operation in HDR mode (i.e., in the event the RSSI again becomes sufficiently low to warrant HDR mode operation), then the host transceiver 112 will switch to channel “L” the next time a transition to MDR mode is required.

[0200] When in the HDR mode and the host transceiver has determined RSSI to be high for K seconds, the mode is switched to the MDR mode. While RSSI is high in the MDR mode, the host transceiver listens for N seconds. If a message is received, the host transceiver stays in the channel in which the message was received. The transceiver remains on the channel provided no interval of greater than M seconds elapses without receiving a message. If after M seconds no message is received and RSSI remains high the communication channel is switched to the next channel and the encryption is reset. Then the host transceiver listens again for N seconds. After N seconds the next channel is selected. After each channel has been selected and no message has been received the transceiver is switched back to the HDR mode and the encryption is reset. In an exemplary embodiment the following values are utilized for the parameters identified in FIG. 17: N=100 ms, M=5 second and K=100 ms.

[0201] Referring to FIG. 18, a flow chart 1800 is provided of an exemplary manner in which a device transceiver 114, 116 transitions between operation within the HDR and MDR modes, and vice-versa. As in the case of the host transceiver 112, a device transceiver 114, 116 transitions from the HDR to MDR mode when an RSSI generated by the applicable RF unit 414, 514 indicates the presence of strong interference (e.g., narrow-band interference).

[0202] As shown in FIG. 18, when the device transceiver 114, 116 is in receiving mode (i.e., in order to receive the ACK), RSSI from the RF unit 414, 514 is checked. If it is determined to be necessary to switch to MDR mode, switching occurs to a particular MDR channel in accordance with the sequence -3-1-4-2-0,... in the case of the initial switch from HDR to MDR mode, the device transceiver 114, 116 first attempts to operate in a predefined MDR channel (e.g., channel 4). If in fact “L” is the channel through which the host/device transceivers are communicating immediately prior to transitioning back to operation in HDR mode (i.e., in the event the RSSI again becomes sufficiently low to warrant HDR mode operation), then the device transceiver 114, 116 will switch to channel “L” the next time a transition to MDR mode is required.

[0203] In an exemplary embodiment the following: values are utilized for the parameters identified in FIG. 18: Timer1=50 ms, Timer2=1 seconds and T=100 ms.

Power-Saving Operation During USB Suspend State

[0204] As was indicated above, in certain embodiments the host transceiver 112 may be incorporated within a dongle or the like and connected to the host PC 102 through a Universal Serial Bus (USB). Those skilled in the art are aware that the USB is a personal computer (PC) interconnect capable of supporting simultaneous attachment of multiple devices. In the exemplary embodiment the host transceiver 112 includes an integrated USB function controller 10 and a full speed (12 Mb/s) transceiver.

[0205] USB devices may be self-powered or receive energy via the USB cable through which they communicate with a hosting entity. USB supports a variety of power modes: On, Suspend, and Off. When placed in Suspend mode, USB devices retain the ability to wakeup the hosting system.

[0206] A number of requirements are imposed upon devices, which are placed on a USB. See, e.g., “An Overview of the Universal Serial Bus (USB), Part 1”, SSS Online (December 2000). For example, a device placed on the bus is permitted to draw up to 500 mA; however, many hosting entities will not permit a device to draw more than 100 mA from the bus. Consistent with the USB protocol, the host transceiver 112 informs the host PC 102 as to the amount of current required for its operation. However, when the host transceiver 112 enters the Suspend mode in accordance with the USB protocol, it cannot draw any more than 500 uA from the bus (assuming the host transceiver is bus-powered rather than battery-powered). In this regard the USB protocol requires the host transceiver 112 to enter the Suspend state if its bus has been inactive for 3 ms. The host PC 102 can initiate a resume command to the host transceiver 112 when it is in Suspend mode. Similarly, the host transceiver 112 can also issue a remote wakeup to the host PC when it is inactive in order to render it active.

[0207] In view of the limitations on the amount of current which can be drawn by the host transceiver 112 from its USB during operation in Suspend mode, it will generally not be possible for the host transceiver 112 to remain “awake” (i.e., fully operational) at all times when operative in this mode. Accordingly, in accordance with one aspect of the invention the host transceiver 112 only periodically becomes fully operational and capable of receiving messages from device transceivers 114, 116 when functioning in Suspend mode. As is discussed below, the messaging protocols of any device transceivers 114, 116 in communication with a host transceiver 112 are also modified upon entry of the host transceiver 112 into Suspend mode.

[0208] Turning now to FIG. 19, a timing diagram 1900 is provided which is representative of the transition of the host transceiver 112 into and out of “awake” and “sleep state” operation during its operation in Suspend mode consistent with the USB protocol. As shown, the host transceiver operates in an awake state for a period of W ms once every B ms, and is otherwise in a sleep state. Each awake period will generally be of the same duration and of sufficient length to permit capturing of the header of a message packet transmitted by a device transceiver. 114, 116. In order to facilitate understanding of this aspect of the invention, exemplary computations of appropriate values of the parameters W and B are given below for the case of transmissions by the mouse transceiver 116 during both normal and spreading mode operation.

[0209] During normal mode operation, the worst-case time required to capture the header of a packet transmitted by the mouse transceiver 116 (i.e., the awake period W) may be expressed as:

\[ W = T_{suc} + T_{ssl} + \text{Packet duration} + T_{tx} + T_{ssl} + \]

ACK time-out duration + T_{ssl} +

acquisition sequence/header duration
In an exemplary implementation of the host transceiver 112, its power consumption during an awake time of 2.190 ms is estimated to be = 44 mW mA. It follows that the interval (B) between awake periods should be <= (44 mW mA)/(2.5 mA), or 17.6 ms. That is, when active in Suspend mode the host transceiver enters an awake state every 17.6 ms (for HDR mode), and is otherwise in sleep state.

During spreading mode operation, the worst-case time required to capture the header of a packet transmitted by the mouse transceiver 116 (i.e., the awake period W) may be expressed as:

\[ W = T_{osc} + T_{PLL} + \text{Packet duration} + T_{tx} + T_{PL} + \]

\[ \text{ACK time-out duration} + T_{PLL} + \text{acquisition sequence/header duration} \]

\[ = (500 + 200 + 573 + 11 + 200 + 253 + 11 + 200 + 253 + 11) \mu s, \] for HDR mode

\[ = 13.0 \text{ ms, for HDR mode} \]

Turning now to FIG. 21, a state transition diagram 2100 depicts the corresponding modifications to the messaging protocol of the mouse transceiver 116 upon entry of its partner host transceiver 112 into Suspend mode. As shown, in this context the mouse transceiver 116 is disposed to re-transmit a packet if an ACK is not received in response to the initial transmission of the packet. If a predefined number of such re-transmissions also fail, the mouse transceiver 116 sends link-reset packets. If these link-reset packets also fail to result in a resetting of the applicable communication link, then the mouse transceiver 116 transmits link-reset packets for 1 ms if no packets are queued for transmission; otherwise, the next queued packet is transmitted and operation continues as described above.

In the embodiment of FIGS. 19-21, the parameters E, B, W, and N are configurable. An exemplary set of default values of these parameters are B=140 ms, W=14 ms, E=140 ms, N=140 ms.

The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that the specific details are not required in order to practice the invention. In other instances, well-known circuits and devices are shown in block diagram form in order to avoid unnecessary distraction from the underlying invention. Thus, the foregoing descriptions of specific embodiments of the present invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, obviously many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated.

While the invention has been particularly shown and described with reference to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made without departing from the spirit and scope of the invention.

What is claimed is:

1. A method for sending messages between computing units of a computing system using radio frequency (RF) communications, comprising:

a) sending a first RF message from a first unit to a second unit and starting an acknowledgement (ACK) timer;

b) receiving said first RF message at said second unit and performing a cyclical redundancy check (CRC) on said first RF message;

c) sending a second RF message from said second unit to the first unit acknowledging that the first message was received when the CRC performed on said first RF message is good;

d) receiving said second message at the first unit and performing the CRC on the second message; and

e) canceling said ACK timer when the CRC performed on the second message is good.
2. The method of claim 1, wherein the first unit is a host computer and the second unit is a peripheral computing device whereby the second unit enters a low power sleep mode upon receiving said second RF message.

3. The method of claim 2, wherein the peripheral computing device is a keyboard, or a mouse, that is RF coupled to the host computer.

4. The method of claim 1, wherein the first unit is a peripheral computing device and said second unit is a host computer whereby the first unit enters into a low power sleep mode upon receiving said second message and performing the CRC to find the second message is good.

5. The method of claim 1, wherein the ACK timer allows a predetermined amount of time for sending and processing of said first and second RF messages.

6. A method for handling a corrupted message sent between computing units of a computing system using radio frequency (RF) communications, comprising:
   a) sending a first RF message from a first unit to a second unit and starting an acknowledgement (ACK) timer at the first unit;
   b) receiving said first RF message at said second unit and performing a cyclical redundancy check (CRC) on said first RF message;
   c) discarding said first message at the second unit when the CRC determines the first message is corrupted;
   d) starting a backoff timer at the first unit when the ACK timer expires;
   e) resending said first RF message from said first unit to said second unit and starting said ACK timer at the first unit;
   f) receiving said resent first RF message at said second unit and performing a cyclical redundancy check (CRC) on said resent first RF message;
   g) sending a second RF message from said second unit to said first unit confirming receipt of said resent first RF message when the CRC on the resent first message is good; and
   h) receiving said second RF message at said first unit, performing the CRC on the second message, and canceling the ACK timer when the CRC on the second message is good.

7. The method of claim 6, wherein the ACK timer allows a predetermined amount of time for sending and processing of said first and second RF messages.

8. The method of claim 6, wherein said backoff timer provides a time delay before resending said RF message thereby breaking deadlocks caused by simultaneous RF transmissions.

9. The method of claim 6, wherein the first unit is a host computer and the second unit is a peripheral computing device whereby said second unit enters into a sleep mode after sending said second RF message and the ACK timer has expired.

10. The method of claim 9, wherein the peripheral computing device is a keyboard, or a mouse, that is RF coupled to the host computer.

11. The method of claim 6, wherein the first unit is a peripheral computing device and said second unit is a host computer whereby said first unit enters a sleep mode after the second RF message is received and CRC is good and the ACK timer expires.

12. A method for handling a corrupted acknowledgement message sent between computing units of a computing system using radio frequency (RF) communications, comprising:
   a) sending an RF message from a first unit to a second unit and starting an acknowledgement (ACK) timer at the first unit;
   b) receiving said RF message at said second unit and performing a cyclical redundancy check (CRC) on said RF message;
   c) sending a first ACK message to said first unit when the CRC determines the RF message is good;
   d) receiving said first ACK message at said first unit, determining said ACK message is corrupt using the CRC, and discarding the first ACK message;
   e) starting a backoff timer at the first unit when the ACK timer expires;
   f) resending said RF message from said first unit to said second unit when the backoff timer expires and starting said ACK timer at the first unit;
   g) receiving said resent RF message at said second unit, determining the resent message good with the CRC, discarding the resent RF message;
   h) sending a second a second ACK message from said second unit to said first unit confirming receipt of said resent RF message; and
   i) receiving said second ACK message at said first unit, performing the CRC on the second ACK message, and canceling the ACK timer when the CRC on the second message is good.

13. The method of claim 12, wherein the ACK timer allows a predetermined amount of time for sending and processing of said RF message and the ACK message.

14. The method of claim 12, wherein said backoff timer provides a time delay before resending said RF message thereby breaking deadlocks caused by simultaneous RF transmissions.

15. The method of claim 12, wherein the first unit is a host computer and the second unit is a peripheral computing device.

16. The method of claim 15, wherein the peripheral computing device is a keyboard, or a mouse, that is RF coupled to the host computer whereby said second unit enters a low power sleep mode after receiving said ACK message with a good CRC.

17. The method of claim 12, wherein the first unit is a peripheral computing device and said second unit is a host computer whereby said first unit enters sleep mode after sending said ACK message.

18. A method for normal radio frequency (RF) communications between a host computer and a peripheral device awakened from a sleep state, comprising:
   a) waking a peripheral computing device from a sleep state by an activated function of said peripheral device;
   b) coupling to a transceiver of said peripheral computing device a signal representing said activated function;
c) sending an RF message representing said activated function to said transceiver of a host computer and starting an acknowledgement (ACK) timer;

e) receiving said RF message by said transceiver of said host computer and determining that the RF message is good by a cyclical redundancy check (CRC);

f) sending an ACK message to said peripheral device indicating said RF message was received; and

h) receiving said ACK message at the transceiver of the peripheral device, determining that the ACK message to be good by the CRC; canceling said ACK timer and returning said transceiver of the peripheral device to said sleep state.

19. The method of claim 18, wherein said peripheral computing devices is a keyboard.

20. The method of claim 18, wherein said peripheral computing device is a mouse.

21. A method for lost or garbled communications between a peripheral device and a host computer using radio frequency (RF) communications, comprising:

a) sending an RF message to a host computer from a peripheral device and starting an acknowledgement (ACK) timer;

b) receiving said RF message by the host computer, determining that a cyclical redundancy check (CRC) of said RF message is not good and discarding said RF message by the host computer;

c) starting a backoff timer when said ACK timer at the peripheral device expires;

d) resending said RF message to the host computer when the backoff timer expires and starting said ACK timer;

e) receiving said RF message at the host computer, determining that the CRC of said RF message is good and sending an ACK message to the peripheral device;

f) receiving at said peripheral device said ACK message and determining the CRC of the ACK message is good; and

g) canceling said ACK timer and placing said peripheral device into a sleep state.

22. The method of claim 21, wherein said backoff timer provides a time delay before resending said RF message thereby breaking deadlocks caused by simultaneous RF transmissions.

23. The method of claim 21, wherein said peripheral device is a keyboard.

24. The method of claim 21, wherein the peripheral device is a mouse.

25. A method for a peripheral device sleep sequence, comprising:

a) awaking a peripheral device when a sleep timer expires;

b) sending a radio frequency (RF) message to the peripheral device to a host computer and starting an acknowledgement (ACK) timer;

c) receiving said RF message at said host computer and determining by a cyclical redundancy check (CRC) that said RF message is good;

d) sending an ACK message to said peripheral device;

e) receiving said ACK message at said peripheral device, determining said ACK message good with said CRC, and canceling said ACK timer; and

f) starting said sleep timer and returning said peripheral device to sleep mode.

26. The method of claim 25, wherein said ACK message has attached to it a command from said host which is processed by said peripheral device if the CRC performed on said ACK message with the command is determined to be good.

27. The method of claim 25, wherein said peripheral device is a keyboard.

28. The method of claim 27, wherein said peripheral device is an optical mouse or a mechanical mouse.

29. A method for a peripheral device sleep sequence with lost messages at the peripheral device, comprising:

a) awaking a peripheral device when a sleep timer expires;

b) sending a radio frequency (RF) message to a host computer from said peripheral device and starting an acknowledgement (ACK) timer at said peripheral device;

c) receiving said RF message at said host computer and determining that a cyclical redundancy check (CRC) of said RF message is good;

d) sending a first host ACK message to said peripheral device;

e) receiving said first host ACK message at said peripheral device wherein said CRC is good and said first host ACK message is discarded;

f) starting a backoff timer when ACK timer at the peripheral device expires;

g) resending said RF message to the host computer when backoff timer expires and starting said ACK timer at the peripheral device;

h) receiving the resent RF message at the host computer, determining the CRC of the resent RF message to be good, ignoring the resent RF message, and sending a second host ACK message to the peripheral device; and

i) receiving said second host ACK message at the peripheral device, determining the CRC to be good, canceling the ACK timer, starting the sleep timer, and placing the peripheral device into a sleep state.

30. The method of claim 29, wherein said backoff timer provides a time delay before resending said RF message thereby breaking deadlocks caused by simultaneous RF transmissions.

31. The method of claim 29, wherein said peripheral device is a keyboard.

32. The method of claim 29, wherein said peripheral devices is a computer mouse.

33. A method for a peripheral device sleep sequence with lost messages at a host computer, comprising:

a) awaking a peripheral device when a sleep timer expires, sending an RF message to the host computer and starting said ACK timer at said peripheral device;
b) receiving said RF message at said host computer,
determining said CRC to be good, sending a host ACK
message with a command to said peripheral device and
starting the ACK timer at the host;
c) receiving said host ACK message with a command at
said peripheral device, determining said CRC of the
host ACK message with said command to be good,
canceling ACK timer at the peripheral device, process-
ing the command, sending a first peripheral device
ACK message to the host and starting the ACK timer;
d) receiving said first peripheral device ACK message at
the host, determining the CRC of the first peripheral
device ACK message to be bad and discarding the first
peripheral device ACK message;
e) starting a backoff timer at said host when the ACK
timer at host expires;
f) resending to said peripheral device the host ACK with
the command when the backoff timer at the host expires
and starting the ACK timer at the host;
g) receiving the resent host ACK with the command at the
peripheral device, determining the CRC of the resent
host ACK with the command to be good, discarding the
resent host ACK as a duplicate of previously received
host ACK message, sending a second peripheral device
ACK message to the host, and starting the ACK timer
at the peripheral device;
h) receiving the second peripheral device ACK message at
the host, determining the CRC of the second peripheral
device ACK message to be good and canceling the
ACK timer at the host; and
i) starting a sleep timer at said peripheral device when
ACK timer at the peripheral device expires and place
peripheral device into a sleep mode.

34. The method of claim 33, wherein said backoff timer
provides a time delay before resending said RF message
thereby breaking deadlocks caused by simultaneous RF
transmissions.

35. The method of claim 33, wherein said peripheral
device is a keyboard.

36. The method of claim 33, wherein said peripheral
devices is a computer mouse.

* * * * *